REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 18.Apr.02	3. REPORT TYPE AND DATES COVERED DISSERTATION

4. TITLE AND SUBTITLE OBJECT RECOGNITION USING AN EXTENDED CONDENSATION FILTER	5. FUNDING NUMBERS
6. AUTHOR(S) MAJ FORBES LANCE A	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) COLORADO STATE UNIVERSITY	8. PERFORMING ORGANIZATION REPORT NUMBER CI02-40
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) THE DEPARTMENT OF THE AIR FORCE AFIT/CIA, BLDG 125 2950 P STREET WPAFB OH 45433	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION AVAILABILITY STATEMENT Unlimited distribution In Accordance With AFI 35-205/AFIT Sup 1	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

20020523 143

14. SUBJECT TERMS			15. NUMBER OF PAGES 208
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

ABSTRACT OF DISSERTATION

OBJECT RECOGNITION USING AN EXTENDED CONDENSATION FILTER

This dissertation describes and tests a set of Condensation Filter extensions that fuse the pose estimates provided by recognition algorithms, while providing a pose localization mechanism to fuse similarity measures from classification algorithms — the traditional Condensation Filter can only accommodate pose estimates. The extensions also fuse the ad hoc information provided by low-level features to serve as heuristics and focus the target pose search. The ad hoc information is combined using the pose of the filter particle as a common reference point and mapping functions derived with Bayesian Learning.

These extensions provide a more flexible framework for better representing and combining diverse sources of information than previously possible. Pose space is also examined more extensively where the conditional target pose probability is higher and more particles are present. This feature gives the Condensation Filter a new active role in directing the use of classification algorithms.

The scaled circular error probable metric (CEP) is used to study the effect on object recognition of combining multiple recognition, classification, and low-level feature extraction algorithms using two different expert combination rules. To examine the relationship between discrimination, accuracy, and precision, receiver operating characteristic curve measurements are compared to the scaled CEP measurements. These results show little correlation between discrimination and accurate and precise localization of the target. The tests performed for this dissertation indicate the Extended Condensation Filter can reliably fuse recognition, classification, and raw feature data to perform more accurate and precise object recognition than is possible using each of the individual contributing algorithms.

Lance A. Forbes
Department of Computer Science
Colorado State University
Fort Collins, Colorado 80523
Fall 2001

To my wife, whose patience and encouragement made this dissertation possible.

Contents

Chapter 1

Introduction

The ability of the Kalman Filter and its variants to provide optimal estimates of a target pose by combining multiple location estimates is well established, where optimal means the filter maximizes the accuracy and precision of the estimate [Kal60, May79, GZ86, HK92, JDM00]. However, for optimal performance the Kalman Filter requires contributing systems provide real-valued linear estimates and the uncertainty associated with the estimates must be reasonably expressed as unbiased additive Gaussian noise [Kal60, May79]. Extensions of the Kalman Filter can accommodate non-linear systems. However, linearization generally requires a hand-crafted solution for each expert and problem domain [May79].

The Condensation Filter accommodates more diverse information than the Kalman Filter [IB98]. For example, the Condensation Filter does not require the uncertainty model associated with a target pose estimate be unbiased or Gaussian. Within computer vision, the term *Condensation Filter* is used to describe a statistical filter with sample-based representation and adaptive resampling. However, the concept has appeared under different names starting over 30 years ago. The ideas of randomly sampling the state of a process, discretely representing the state distribution, and adaptively resampling can be found in the Monte Carlo Filter described by Handschin and Mayne [HM69]. Rubin's sample-importance-resample (SIR) work provides a practical approach and justification for adaptive resampling [Rub87]. The first complete description of the Condensation Filter, called the Bootstrap Filter, is found in Gordon's, Salmond, and Smith's paper on nonlinear non-Gaussian state estimation [GSS93, Gor97]. Application of the Condensation Filter to computer vision is described by Isard and Blake [IB98], where the name Condensation Filter is first used.

DISSERTATION

OBJECT RECOGNITION USING AN EXTENDED CONDENSATION FILTER

Submitted by
Lance A. Forbes
Department of Computer Science

In partial fulfillment of the requirements
for the Degree of Doctor of Philosophy
Colorado State University
Fort Collins, CO
Fall 2001

COLORADO STATE UNIVERSITY

September 6, 2001

WE HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER OUR SUPERVISION BY LANCE A. FORBES ENTITLED OBJECT RECOGNITION USING AN EXTENDED CONDENSATION FILTER BE ACCEPTED AS FULFILLING IN PART REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Committee on Graduate Work

Adviser

Department Head

Despite the Condensation Filter's less restrictive assumptions, it cannot accommodate a large amount of the information available from many computer vision algorithms. Many algorithms, such as classifiers and low-level feature extractors, do not provide an estimate of the target object pose. Classifiers provide a measure of similarity between two images, one of which is usually the target. Low-level feature extractors provide a set of features generally assumed to have some relationship to the target. Results in this dissertation show both classifiers and low-level feature extractors can provide information to improve the object recognition process. However, until this point, the Condensation Filter has been unable to use such information.

To illustrate the distinction between these different types of information, assume the goal is estimating, with the greatest accuracy and precision possible, the two-dimensional pose of an object. If two algorithms each produce their own direct estimate of the two-dimensional pose, fusing information requires manipulation of uncertainty bounds, but translation from one representation to another is not required. In contrast, assume one of the algorithms provides a direct estimate of position, and another algorithm provides the temperature at different positions. Temperature may be a very useful indirect cue, but a mapping must be established between temperature and object position before the two types of information can be combined.

This dissertation presents two novel extensions for the Condensation Filter to make use of both direct and indirect sources of visual information in the context of recognizing known objects in video sequences. The first contribution is an extended architecture that recognizes and accommodates three distinct types of information sources: recognition algorithms, classification algorithms, and low-level feature algorithms. Recognition algorithms provide a pose estimate and possibly a similarity score. Given two images to compare, classification algorithms provide only a similarity score and they include no pose localization mechanism. Low-level features, such as temperature, color, or range, provide only vague cues indicating where an object may be located.

To implement this first extension, each particle within the Extended Condensation Filter is used as a common reference point for combining the diverse information provided by direct and indirect sources. To accommodate recognition algorithms, each particle pose is compared to each recognition algorithm pose estimate. Much like a traditional Condensation Filter, the distance between the particle pose and the estimate pose is used to estimate the probability the particle pose is near the target object. To accommodate classifiers, the particle pose is used to create a transformation matrix and render an estimate of the target object appearance. This estimated target appearance allows a classifier to compare the target to the observed image and estimate the probability the particle pose is near the target pose. To accommodate low-level feature extractors, the particle pose is compared to each extractor's set of low-level features to estimate the probability the particle pose is near the target pose. For each type of information, the common point of comparison is the particle pose and the common output is the conditional probability the particle pose is near the true target pose.

The focus on the particle pose gives the Extended Condensation Filter a more active role than the traditional Condensation Filter. Because classification similarity measurements are only made at particle poses and the filter purposely moves particles in pose space, the filter actively directs the application of the classifiers. This represents a significant departure from the traditionally passive role of the Condensation Filter, where observations are received and used to move particles within the filter, but the position of the particles has no influence on the application of the contributing sources of information.

The second contribution applies Bayesian learning to characterize and normalize the conditional probability models for each information source combined within the Condensation Filter. These conditional probabilities provide a common representation of the uncertainty associated with direct and indirect sources of information. Algorithms based on edge matching, color index, and stereo disparity provide very different indications of object position and similarity. This dissertation provides a method for comparing and combining this information using learned probabilities.

When proposing a new algorithm, one naturally desires a comparison with the current standard. For the reasons discussed earlier, the Kalman Filter and traditional Condensation Filter do not provide a comparable capability because they only accomodate a subset of the information accommodated by my extensions. There are systems, such as a Bayesian Network, which can combine direct and indirect sources of information [Pea88]. However, a Bayesian Network requires the specific semantics of the output of each expert be encoded by the user into the network architecture, while my system uses a single general architecture. As a result, there is no basis for comparison with the system I propose.

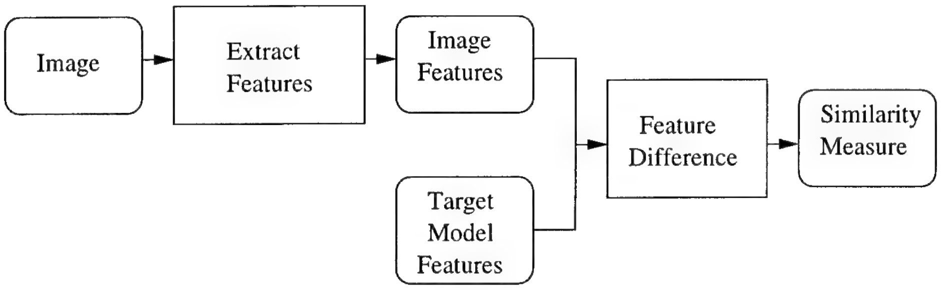


Figure 1.1: Classification

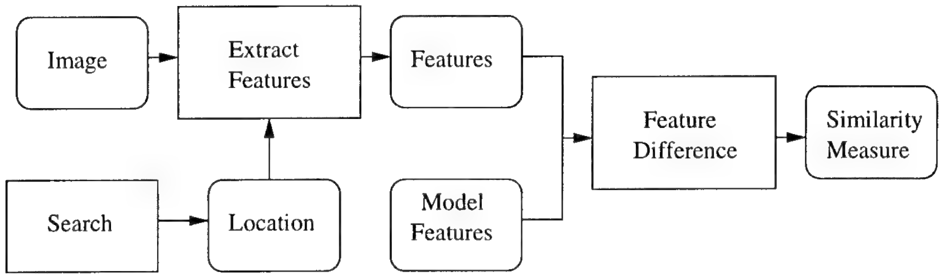


Figure 1.2: Adding Location to Classification by Transforming Image Features to Match the Model

1.1 Recognition = Classification + Localization

Object recognition contains two sub-problems: object classification and object localization [TV98]. Classification measures the similarity between two images, but performs no localization. As shown in Figure 1.1, object classification requires a minimum of two sets of features; at least one set of features extracted from the scene and one set of features for each target model. The image features and target model features are compared using a similarity measure. For example, the target and image features may be represented as two vectors and the Euclidean distance between the feature vectors represents their similarity — the smaller the distance, the greater their similarity. Assignment of a set of features to a specific class is usually based on comparing the similarity measure to a threshold.

Object localization is also required to perform object recognition. As defined by Trucco and Verri, object localization requires the estimation of the three rotation dimensions and three translation dimensions (pose) of an unknown object in pose space [TV98]. For this dissertation, this pose is the vector $\vec{s}$. Other methods for finding the pose estimate, such as the generalized Hough transform, perform search in correspondence space [Bal81, Bev93]. More recent research has explored the utility of searching directly in pose space [HR92,

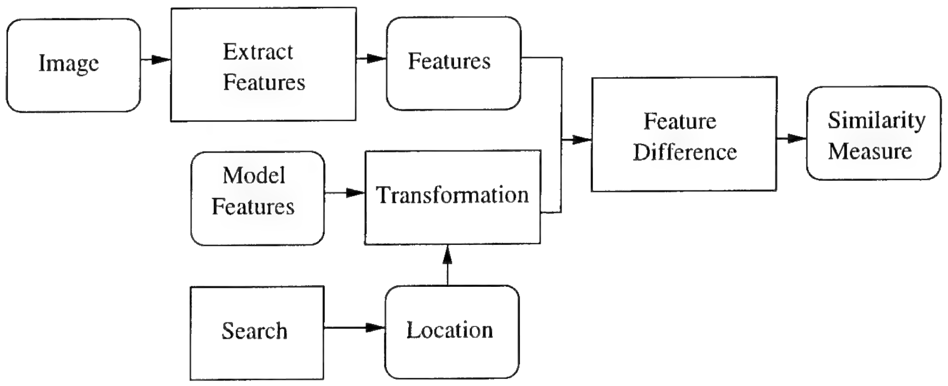


Figure 1.3: Adding Location to Classification by Transforming the Model Features to Match the Image

RB95, SB01]. When searching in pose space, any object classification algorithm can be used to estimate the pose of the target object in an arbitrary image by computing the similarity measure between the target model features and image features over a range of hypothetical poses. Features can be extracted from the image based on a hypothetical pose and compared to the target model features, as shown in Figure 1.2, or the target model features can be projected to the hypothetical image pose, as shown in Figure 1.3. With either method, the pose maximizing similarity is assumed to be the most likely.

The process of projecting and comparing can be used to search the image for instances of the target class, but the search can be computationally demanding. Some object recognition algorithms have specialized heuristics that speed-up pose localization by skipping obviously unattractive portions of the search space [HR92, RB95], but they still must search in pose space using the classifier similarity measure as the objective function. The search space for all but the most simple images is very challenging and specialized search heuristics have only been developed for a small number of classification algorithms. As a result, most classification algorithms must exhaustively search to predictably and reliably perform object localization.

1.2 Extending the Condensation Filter

If several recognition algorithms are combined to provide a single estimate, such as the product of a Kalman Filter, the combined estimate could be used to direct the search to the most likely pose. Unfortunately, the Kalman Filter does not provide a general purpose

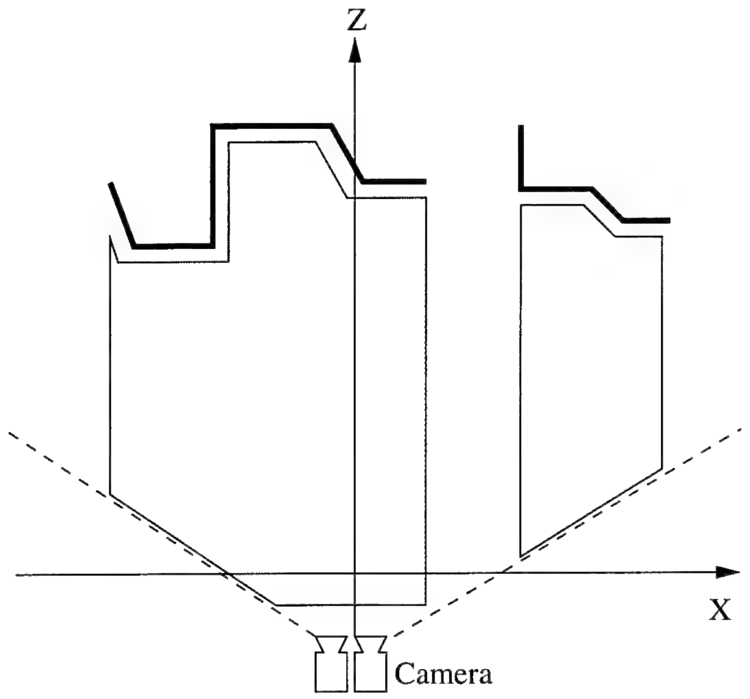


Figure 1.4: The range features are shown as a bold line. The pose space eliminated by low-level range features is shown in gray.

search mechanism, so a separate process would need to perform the search. Fortunately, the Condensation Filter contains a general purpose search mechanism similar to an Evolution Strategy [Fog00], and it has much less restrictive assumptions than the Kalman Filter. The relationship between Evolution Strategies and the Condensation Filter will be discussed in more detail later.

The Condensation Filter is a recent development claiming statistical filtering properties similar to the Kalman Filter — some of these claims are questioned later in this dissertation. However, in contrast to the Kalman Filter, the underlying mechanism of the Condensation Filter is similar to population-based search. As a result, the Condensation Filter can be extended to provide the statistical filter required to fuse pose estimates from object recognition algorithms while providing the pose search required to fuse the similarity measure provided by object classification algorithms. To accommodate classification algorithms, the filter particle is used as a common point of comparison to create a classification problem. The result is a single process capable of fusing results from any recognition algorithm that provides a target object pose estimate and any classification algorithm that provides a similarity measure.

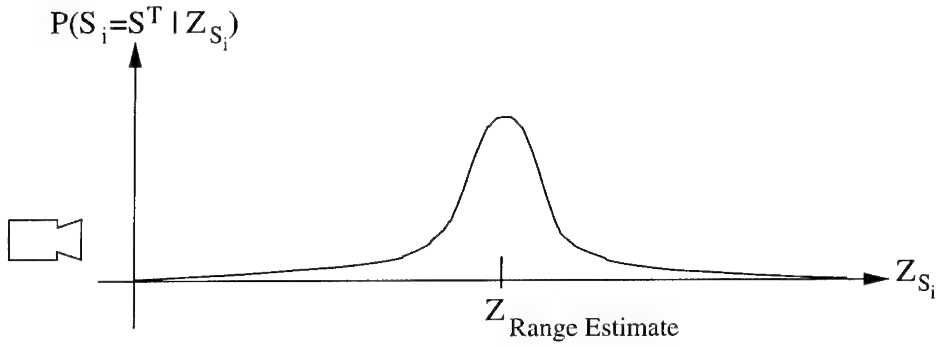


Figure 1.5: A cross-section of Figure 1.4 showing a probabilistic error model for a single range feature

In addition to fusing classification and recognition algorithm information, my Condensation Filter extensions allow fusion of low-level feature data. Low-level features do not provide a location estimate nor a similarity measure, but they can help prune the object pose search space. For example, the low-level range information provided by a stereo disparity algorithm does not provide a similarity measure nor an object pose estimate. As shown by the example in Figure 1.4, the low-level range data only estimates the nearest line-of-sight distance to objects in the scene. However, these simple features indicate that the target object is not in the portion of the pose space shown in gray because the nearest line-of-sight object detected by the stereo disparity algorithm is the bold black line.

The discrete representation (gray = no object, black = object) in Figure 1.4 is a simplification. Stereo disparity algorithms make mistakes and, as a result, may incorrectly estimate range, so there is a finite probability that the object is closer or further away than the feature indicates. Just like fusing pose estimates in a Kalman Filter, fusing low-level feature data in a Condensation Filter benefits from accurate error models. For example, a stereo disparity feature error model, like the one shown in Figure 1.5, estimates the probability a pose represents an object surface based on the difference between the range to a pose and the stereo disparity feature. Results reported later in this dissertation indicate feature extractor error models can be difficult to represent parametrically and are highly dependent on the task, so my extensions must provide a flexible error model representation. My Condensation Filter extensions accommodate the diverse information and error models provided by low-level features, classification similarity measures, and recognition pose estimates.

1.3 Example

To illustrate the practical need for a system capable of fusing the results from recognition, classification, and low-level features extractors, consider the following example. Your goal is to design a system to help the police visually locate a vehicle parked along the side of the road. Your system will traverse long segments of road looking for the target vehicle based on the available target vehicle feature data. Navigation will be performed by a separate system — possibly a police officer. To minimize the waste of law enforcement resources your system should only generate an alert when a probable match is found, by reliably identifying probable target vehicles — this requires good target discrimination. Given a variety of parking options, the target vehicle might be viewed with any ground-plane rotation. Ground-plane translation of the target vehicle is limited only by the field of view and resolution of the camera. Only small vertical translations are likely because the grade of a road is typically limited to relatively low inclines compared to the range of the camera. The police patrol vehicle may traverse roads in both directions, so multiple distinctly different views of every car may be available.

Unfortunately, fleeing criminals are frequently uncooperative, so the police and your system can only be sure of having one or two images of the target vehicle prior to the search. This limited *a priori* data usually rules out appearance-based methods such as Eigenspace manifolds [Bor99]. Fortunately, several algorithms that perform object recognition with minimal *a priori* data are available. A color histogram could be used to look for objects of the vehicle's color, but common vehicle colors, such as white, will be a problem [SB91]. Edge template matching could be used, but vehicles of the correct model and the wrong color will report as false matches [HR92]. Edge templates may also miss discolorations on the vehicle. Pixel-level statistical correlation might work, but specialized hardware will be required to perform the exhaustive search necessary to localize the pose of the target object at frame rate [SB01]. Statistical correlation may also fail in the presence of high out-of-plane rotation. In short, every algorithm based on a single set of features has specific and predictable strengths and weaknesses.

Multiple views of each vehicle will be available as the camera travels along the road. Each view offers a unique opportunity to match the image to the target model because different lighting, background clutter, and occlusion will be present in each camera view.

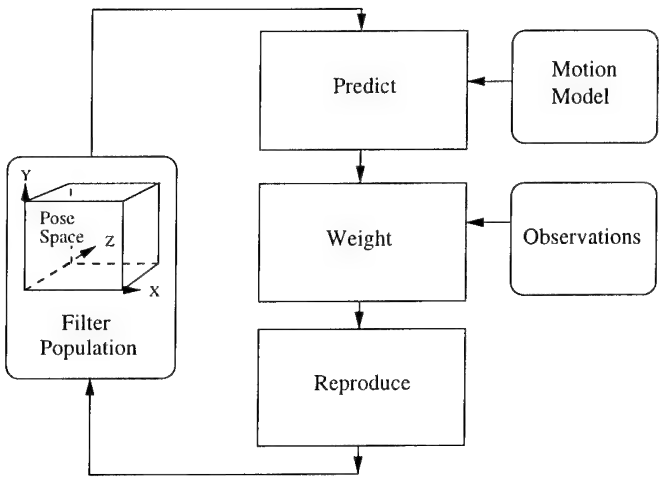


Figure 1.6: Top-level architecture of the canonical Condensation Filter

However, each view will have to be registered in a common reference frame. In theory, if the multiple views could be processed by multiple object recognition algorithms then fused, the resulting object recognition estimate should be superior to any of the contributing estimates. If low-level feature data and object classification results could also be fused with the recognition estimates, object recognition should further improve. Ideally, a diverse set of recognition, classification, and low-level feature extraction algorithms should be used to process every image frame. The result should be a set of target object pose estimates represented in a single reference frame that are more discriminating, accurate, and precise than any of the contributing algorithms.

This vehicle search problem will be the basis for the test problems used later in this dissertation because it contains the key problems of combining recognition, classification, and low-level feature data. It also provides real-world complications such as combining estimates from different spatial reference frames, limited *a priori* data, and noisy and heterogeneous features and pose estimates.

1.4 Research Focus

The architecture of the canonical Condensation Filter is shown in Figure 1.6 [IB98, $D^{+}99$, Mac00]. The central data structure of the Condensation Filter is a fixed-size population of particles, as shown at the top of Figure 1.7, where each particle represents a hypothetical target object pose. The fundamental steps used to update the filter are: predict pose changes

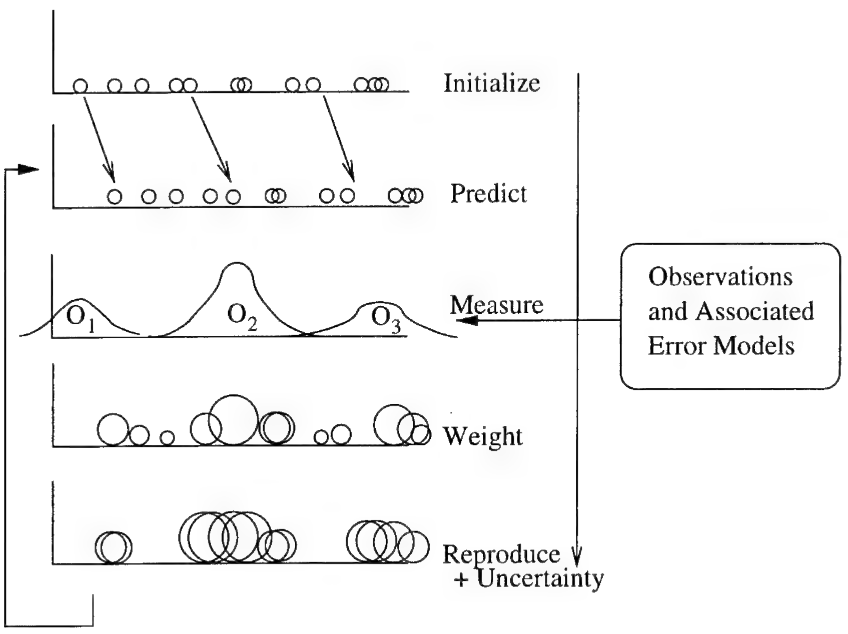


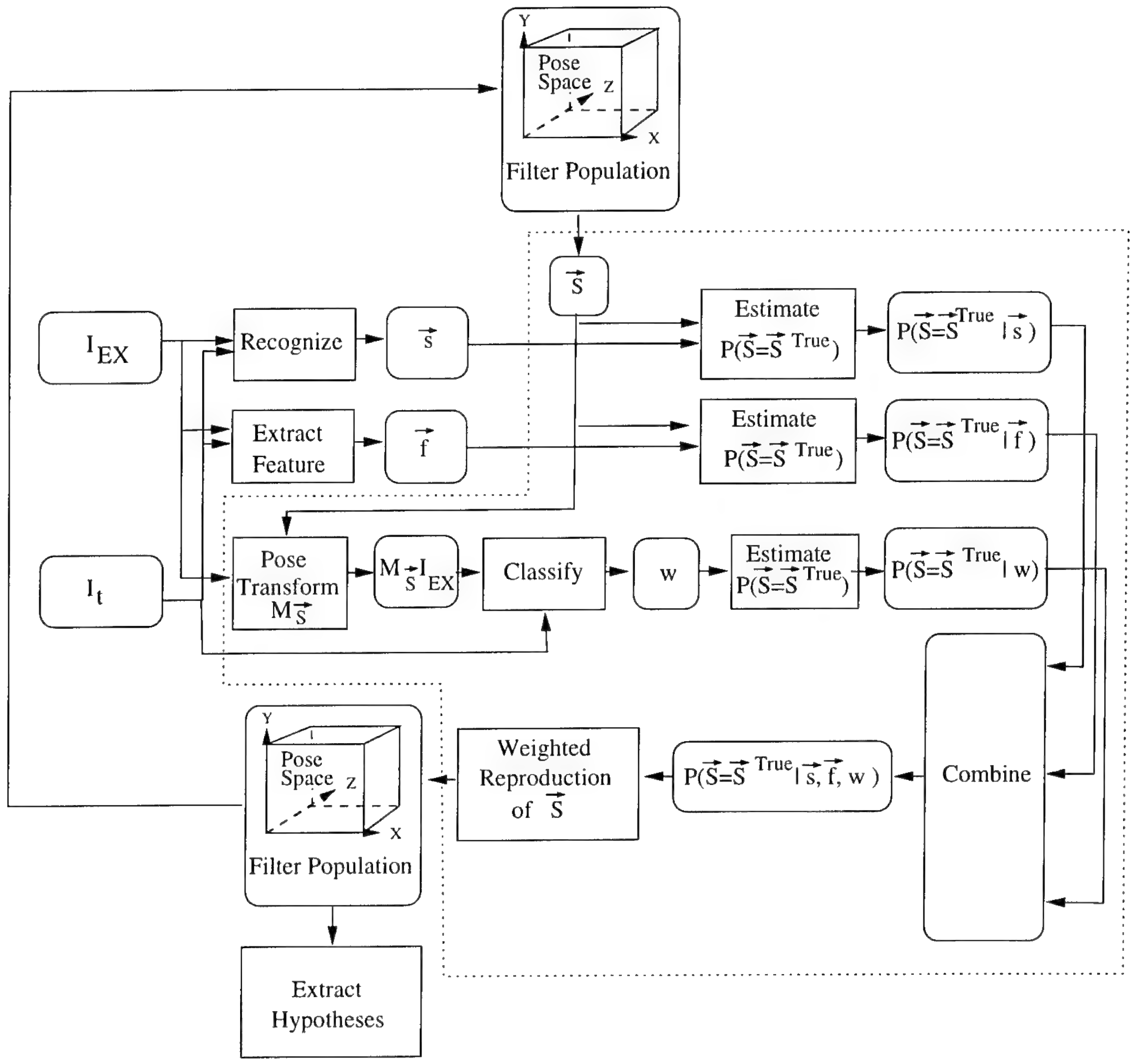
Figure 1.7: Top-level architecture of the canonical Condensation Filter showing a small population of particles

using the target object's motion model, weight the particles in the Condensation Filter based on the observations, and reproduce the particles in the Condensation Filter in proportion to their weight. As shown at the bottom of Figure 1.7, the predict-weight-reproduce cycle causes the particles in the filter to move (or condense) toward the observations. More about the theory and implementation of the Condensation Filter will be discussed in Chapter 2.

Figure 1.8 illustrates the Condensation Filter extensions that are the focus of this dissertation. To make the filter extensions more obvious, Figure 1.8 shows only the operations required to process a single filter particle based on a single acquired image. The weighting operations inside the dotted line must be repeated for each particle in the filter. Each new image also requires the prediction operation shown in Figure 1.6, but not shown in Figure 1.8. To clarify the relationship between Figure 1.6 and 1.8, place the operations inside the dotted line of Figure 1.8 within a loop that increments from particle one through the entire population of particles, then use the new loop to replace the *weight* and *reproduce* boxes in Figure 1.6.

As shown in Figure 1.9, which highlights the basic stages of Figure 1.8, all the operations in Figure 1.8 can be reduced to four primary stages

One Obtain expert estimates



Pose Space
Y
Z
X
Filter Population
$\vec{S}$
I_{EX}
Recognize
$\vec{s}$
Extract Feature
$\vec{f}$
Estimate $P(\vec{S}=\vec{S}^{True})$
$P(\vec{S}=\vec{S}^{True} \mid \vec{s})$
Estimate $P(\vec{S}=\vec{S}^{True})$
$P(\vec{S}=\vec{S}^{True} \mid \vec{f})$
I_t
Pose Transform $M_{\vec{S}}$
$M_{\vec{S}} I_{EX}$
Classify
w
Estimate $P(\vec{S}=\vec{S}^{True})$
$P(\vec{S}=\vec{S}^{True} \mid w)$
Combine
$P(\vec{S}=\vec{S}^{True} \mid \vec{s}, \vec{f}, w)$
Weighted Reproduction of $\vec{S}$
Pose Space
Y
Z
X
Filter Population
Extract Hypotheses

Figure 1.8: Operations performed by the Extended Condensation Filter to weight a single particle based on the particle's pose estimate. The operations inside the dotted line are repeated for each particle in the filter. I_{EX} is the target object example image. I_t is an unknown image acquired at time t. $\vec{s}$ is a target pose estimate provided by a recognition algorithm. $\vec{f}$ is a feature provided by a low-level feature extractor. $\vec{S}$ is a pose assigned to a single particle in the filter's population of particles. $M_{\vec{S}} I_{EX}$ is the target image transformed on the basis of the pose $\vec{S}$. w is the classification algorithm's measurement of similarity between the transformed target image $M_{\vec{S}} I_{EX}$ and the acquired image I_t. $\vec{S}^{True}$ is the true target pose, and $P(\vec{S} = \vec{S}_{True})$ is the probability that $\vec{S}$ is the true target pose.

Two Combine independent estimates

Three Weighted random reproduction

Four Extracting hypotheses

All four stages offer potentially rewarding research questions, but I have elected to focus on the second stage — combining data sources. The next several sections describe each of these stages in detail and my reasons for focusing on the process of combining independent estimates. A detailed pseudo-code description of this algorithm is provided in Chapter 3.

1.4.1 Stage One - Feature Extraction

Stage One in Figure 1.9 extracts features from the acquired image. These feature sources will directly affect the performance of the combined recognition algorithm — as discussed at the beginning of this chapter. New and different types of edge detectors, color histogram algorithms, classifiers, and recognition algorithms are presented at nearly every computer vision conference. Their strengths and weaknesses are frequently debated and each could be plugged into the front end of my filter and tested. However, I want to demonstrate the generality of my Extended Condensation Filter and its ability to fuse data from diverse recognition, classification, and low-level feature extraction algorithms, rather than demonstrate optimal performance in some problem domain. For this reason I have emphasized diversity rather than maximum performance and selected an edge-based recognition algorithm, a color-based classifier, a pixel-correlation-based classifier, a stereo-disparity range feature extractor, and a color-histogram back-projection algorithm [HR92, SB91, SB01, DH73]. The role of each of these algorithms will be discussed in Chapter 3.

1.4.2 Stage Two - Combining Data Sources

Stage Two in Figure 1.9 combines the features extracted in Stage One. The features ($\vec{S}$, $\vec{f}$, and w) extracted in Stage One provide diverse information about any hypothetical pose. For a population-based algorithm, this diverse information must be used to evaluate the fitness of every member, or particle, in the population. As a result, every feature must be converted to a particle-specific weight. The weight represents the likelihood that a particle is the correct estimate of the target position as compared to the other particles in the population. This stage is the key point in the Extended Condensation Filter that allows

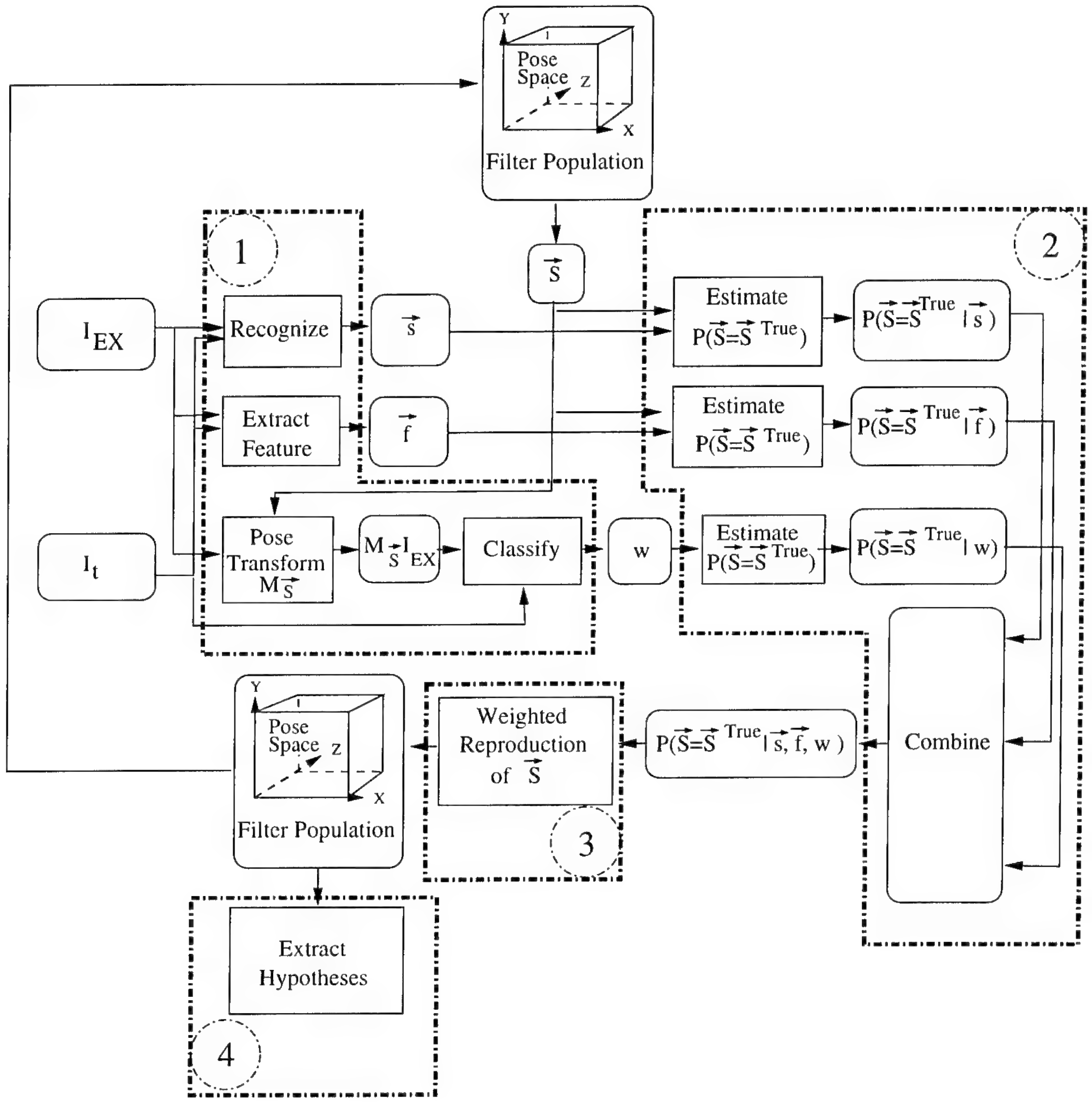

Pose Space
Filter Population
1
2
3
4
I_{EX}
I_t
Recognize
Extract Feature
Pose Transform $M_{\vec{S}}$
$M_{\vec{S}} I_{EX}$
Classify
$\vec{S}$
$\vec{s}$
$\vec{f}$
w
Estimate $P(\vec{S}=\vec{S}_{True})$
$P(\vec{S}=\vec{S}_{True} | \vec{s})$
$P(\vec{S}=\vec{S}_{True} | \vec{f})$
$P(\vec{S}=\vec{S}_{True} | w)$
Combine
$P(\vec{S}=\vec{S}_{True} | \vec{s}, \vec{f}, w)$
Weighted Reproduction of $\vec{S}$
Extract Hypotheses

Figure 1.9: Fundamental stages within the Condensation Filter extensions: 1 - Independent Expert Estimates, 2 - Combining Independent Estimates, 3 - Weighted Reproduction, 4 - Extract Hypotheses.

recognition, classification, and low-level features to be combined and it directly affects the generality and performance of the filter. Stage Two is the primary focus of this dissertation.

1.4.3 Stage Three - Weighted Reproduction

Stage Three in Figure 1.9 reproduces the particles in the population in proportion to the weight assigned in Stage Two. The weighted reproduction stage is similar to weighted reproduction in an Evolution Strategy [DL00]. As a result, nearly all the research on weighted reproduction in population-based search algorithms would apply to the weighted reproduction of population-based filters. Unfortunately, I do not have the time to adequately address the wealth of advanced techniques that could be borrowed from the search literature and applied at this stage. I have elected to use the canonical filter reproduction algorithm, where the probability of reproducing particle S_i is the ratio of the weight for particle S_i divided by the total weight for all particles in the population

$$P(reproduction) = \frac{w_i}{\sum_{i=1}^{N} w_i} \tag{1.1}$$

where $P(R_i)$ is the probability of reproducing particle i, and w_i is the weight of particle i in a population of size N. When used for an Evolution Strategy, this formulation is called *fitness proportional* reproduction [Fog00]. The details of particle reproduction will be discussed further in the Chapters 2 and 3.

1.4.4 Stage Four - Extract Hypotheses

Group Four in Figure 1.9 extracts target pose hypotheses from the state of the filter particles. In many applications of the Condensation Filter this is treated as a trivial operation – the mean pose of all the particle poses is assumed to be the hypothesis with the best support. This is a crude and potentially incorrect assumption. One advantage of the Condensation Filter is its ability to represent ill-defined multi-modal distributions. When multiple target objects or similar false target objects are present, multiple modes may be represented by the particles in the filter. For this dissertation, distribution assumption are avoided by using the distribution independent circular error probable (CEP) statistic to measure the accuracy of the filter state in total. In addition, clustering is used to test the validity of the single-mode assumption. These tests will be discussed further in Chapter 4.

In summary, there are four major areas for research regarding my Extended Condensation Filter; feature extraction, classifier combination, weighted reproduction, and hypothesis extraction. For stage one, research on feature extraction is intentionally limited by selecting an arbitrary, but diverse, group of features that will demonstrate the Extended Condensation Filter's generality. Stage two, classifier combination, will be the primary focus of this dissertation. Stage three, weighted reproduction, will use the reproduction algorithm from the canonical Condensation Filter. Finally, for stage four, assumptions about the filter population are avoided by using the distribution independent CEP statistic.

1.5 Dissertation Overview

This dissertation is composed of six chapters. The remaining chapters are:

Background The Condensation Filter extensions described and tested in this dissertation rely on previous research in several areas. Chapter 2 reviews the most relevant research with a focus on feature extraction, combining classifiers, localization, and Evolution Strategies.

Theory Chapter 3 describes several abstract models of the Extended Condensation Filter and the associated theory. Chapter 3 also poses a set of research questions prompted by the models.

Test Chapter 4 describes one implementation of the Extended Condensation Filter, a set of controlled test environments, and a set of empirically testable hypotheses derived from the research questions presented in the previous chapter. These tools are used to generate the empirical test results reported in the next chapter.

Results Chapter 4 summarizes the results of testing the Extended Condensation Filter implementation described in the previous chapter. This chapter also draw some conclusions regarding the hypotheses and research questions posed earlier.

Conclusions Chapter 6 draws some general conclusions regarding the Extended Condensation Filter and outlines future research topics.

Chapter 2

Background

A statistical filter, such as the Kalman Filter, can seem magical to the uninitiated. Adding an extremely bad estimate to an extremely good estimate does not degrade the good estimate. Instead, given a well designed filter, the combined estimate is better than each of the individual contributing estimates. The result seems counter-intuitive. Putting a bad human navigator in a car with a good human navigator usually confuses the driver. The bad navigator may convince the driver to take the occasional wrong turn. But statistical filters assume the properties of the good and bad navigator are well understood — each is represented by a statistical error model. In a sense, our driver knows when the good navigator is right and when the bad navigator is wrong. This information is essential for a statistical filter's "magic."

Parametric statistical filters have a long history. Emil Kalman's first paper on the Kalman Filter was published over 40 years ago, and many extensions are well established [Kal60, May79]. The Kalman Filter and its extensions are commonly used in aerospace navigation systems and ground tracking systems to combine diverse sources of object location information. However, the limiting assumptions of parametric statistical filters make them difficult to apply to visual object recognition.

In general, the product of visual object recognition algorithms may not fit the equations required by parametric filters. Other algorithms, such as classifiers and low-level feature extractors, do not provide an estimate at all. Classifiers, for example, only measure similarity. Using the previous good-bad navigator example, a classifier is like a navigator who only "knows it when he gets there." A low-level feature extractor only looks for some specific set of features. This is like a navigator who only knows the destination is painted white. Such information may prove useless to the driver who requires left and right turn information.

When I started this research, my goal was a Kalman Filter for visual object recognition. I wanted the filter to accommodate as many sources of object recognition information as possible, as well as perform object recognition in dynamic environments. I also wanted the "magical" property of combining well-modeled diverse sources to create recognition estimates better than each of the individual contributing sources. Unfortunately, the Kalman Filter's parametric assumptions fail to support many, if not most, of the potentially useful sources of information for object recognition.

To accommodate recognizers, classifiers, and low-level feature extractors with arbitrary uncertainty models, I have extended the Condensation Filter. The canonical Condensation Filter provides a very flexible non-parametric representation of the combined target pose estimate and associated uncertainty. However, the Condensation Filter is still too limited, because it requires parametric representations of the uncertainty for each contributing source. The Condensation Filter also has no mechanism to make use of classification or low-level feature information. My extensions allow the filter to learn a discrete representation of the uncertainty for each contributing information source and compare classification and low-level features to pose estimates. This dissertation describes and tests these extensions.

Computer science, statistics, and mathematics, among other fields, contribute to my work, but it would be impractical to cover all these contributions in detail. The practical, but limited, goal of this chapter is providing the background necessary to discuss the theoretical models, research questions, and testable hypotheses in later chapters. The first section of this chapter describes a limited taxonomy of object recognition algorithms. This taxonomy will help describe the feature sets used to test the Condensation Filter extensions. The second and third sections briefly review classifiers and combining classifiers. Two of the classifier combination algorithms discussed will be tested in later chapters. The fourth and fifth sections review object recognition and the more popular algorithms for combining pose estimates. These sections focus on the strengths and weaknesses of the Kalman Filter, Mixture of Gaussians Filter, and Condensation Filter. Section six reviews population-based search, with a focus on Evolution Strategies and their similarity to the Condensation Filter.

2.1 Features

One of the primary goals of my research, established in the next chapter, is improved object recognition through the fusion of diverse sources of information — specifically low-level features, similarity measures, and pose estimates. This section argues that diverse algorithms provide distinctly different advantages, depending on the problem domain. The car search example in Chapter 1 illustrates how combining diverse information can provide an advantage in practical problems.

To demonstrate that a new algorithm accommodates diverse algorithms, it should be demonstrated using diverse algorithms. However, proving a group of object recognition algorithms is diverse is difficult, if not impossible. If the algorithms could be described by a scalar, algorithms from each end of the scale could be selected and reasonably called diverse. Unfortunately, the features, similarity measures, and localization algorithms within the extensive literature on object recognition are much too complex and numerous to describe by a simple scalar. In fact, no taxonomy of object recognition algorithms is well-established in the computer vision literature. Rather than abandon the notion of selecting diverse sources of information, this chapter describes a small sub-set of the more obvious and well-established dimensions of object recognition algorithms. Using a small sub-set of dimensions as the basis for selecting features and similarity measures cannot guarantee a representation of the spectrum of object recognition algorithms, but selection on the basis of these dimensions will ensure my algorithms are dissimilar.

2.1.1 Low-Level Versus High-Level Features

The level of feature abstraction is one of the most common ways to differentiate sources of object recognition information. Low-level features, such as edges, color, and depth, are extracted from the image pixel data. Classification similarity metrics are extracted by comparing low-level features extracted from two images. Target pose estimates are typically based on comparing many classification similarity metrics. Each step in this process represents a shift to a higher level of abstraction requiring additional computation. Each level of abstraction also loses some information about the original image.

In terms of abstraction, low-level features are close to the original pixel data and are not very selective about the information they encode. From the standpoint of an object recognition task, the information in low-level features is likely to be irrelevant. In contrast, high-level features are typically farther removed from the pixel data and contain less irrelevant detail. However, in practice it is nearly impossible to design general feature extractors that perfectly discard the irrelevant detail and retain the relevant information. Thus, some relevant information is usually lost in the abstraction process.

Features can be measured on a scale from low to high. The original image sits at the low extreme of this scale and object recognition sits toward the high end, while low-level features and classification sit somewhere in-between. For example, in the object recognition process shown in Figure 1.3 on page 5, the original unknown image contains all the information the sensor collected. Features, such as edges, extracted from the original image contain only the information relevant to the computation of a similarity metric. A classifier, such as the Hausdorff algorithm described later, may provide only a single metric measuring the similarity of the unknown image features to the target model features. When the similarity metric is used to perform object recognition, the classifier may be run repeatedly to provide many similarity metrics, but the recognition algorithm may provide only a single target object pose estimate. Each step from low-level to high-level provides less, but increasingly relevant, information.

2.1.2 Object-Centered Versus Viewer-Centered

Object recognition algorithms can be classified as object-centered or viewer-centered [Mar82]. Object-centered methods use feature sets that may not be obvious to the viewer. For example, a list of the geometric vertices of the target object is an object-centered representation. Viewer-centered models describe the appearance of the object from a viewer's point of view. One example is the eigenspace representation that combines multiple images of the object from the viewer's perspective. While object-centered methods use models to build a view of the object, viewer-centered methods use object views to build a model. Each method has predictable strengths and weaknesses. For example, object-centered methods are dependent on the accurate extraction of model features, such as edges or corners. Viewer-centered methods are dependent on proper segmentation of the target from the background and lighting.

2.1.3 Local Versus Global Features

Object recognition methods can also be classified as local or global. Local methods attempt object recognition by finding some subset of features that belong to the target object, while global methods search for the target object in its entirety. Pixel correlation can be used to implement both local and global algorithms. A local implementation of pixel correlation measures correlation between patches of the target object image and the sensed image. The local method has the advantage of detecting parts of the target object even when the target object is partially occluded. Unfortunately, the local method may produce a false positive match when a non-target object is composed of pieces similar to the target object, but assembled differently. A global implementation of pixel correlation measures correlation between the entire target image and the acquired image, but it is sensitive to occlusion and the properties of the background. Hybrid methods can incorporate some of the strengths of both methods.

2.1.4 Radiometric Versus Geometric Features

Recognition algorithms can use either radiometric or geometric features. Geometric feature extraction infers structure. Radiometric feature extraction infers material surface properties.

Radiometric methods infer the surface properties of objects in the scene, such as the illuminant response, from the pixel intensity values. A three-dimensional red-green-blue (RGB) histogram is a purely radiometric feature set [SB91]. The RGB value of every pixel in the image is used to create a three-dimensional histogram that represents the number of times every distinct color in the image occurred. The histogram contains a great deal of information about the radiometric properties of the image, but says nothing about the geometric arrangement of those colors. If pairs of pixels in the original image were randomly swapped, the histogram would be unchanged. Because pixel values can only be used to infer surface properties, radiometric methods are more sensitive to lighting changes than geometric methods. If a white cube is placed under a blue light, it appears blue. If the same white cube is placed under a red light, it appears red. The RGB histogram for each of these two images appear completely different despite the presence of the same object. The RGB histogram will be used in two variations to test the Extended Condensation Filter.

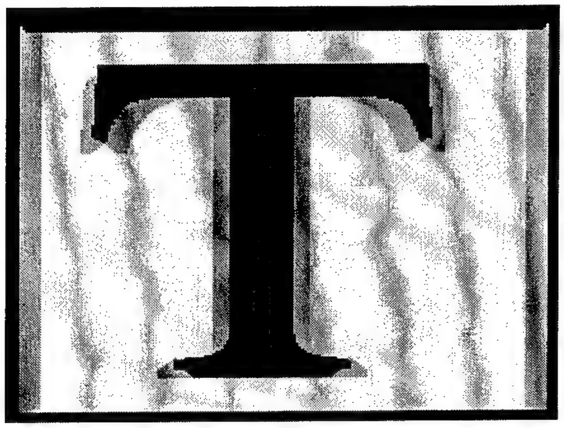


Figure 2.1: Original image

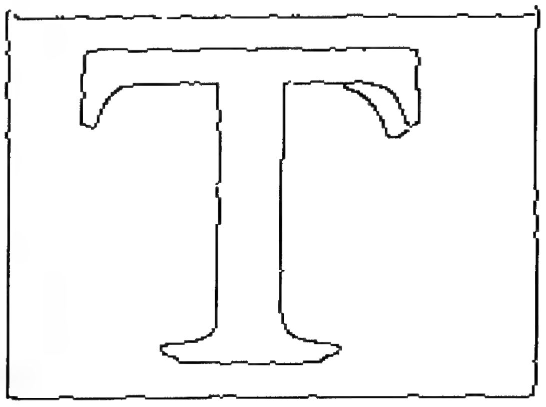


Figure 2.2: Geometric edge features extracted by Canny

Geometric methods infer structure from the pixel values and try to classify an object from the structure. Geometric methods, such as model-based object recognition, first extract local geometric features, such as edges or lines. Classification or recognition compares these geometric features.

Edge patterns are an example of purely geometric features. The Canny edge detector extracts edge features from an image based on image intensity gradients [Can86]. Depending on the parameter settings, Canny may extract the edge features shown in Figure 2.2 from the image shown in Figure 2.1. Notice that the output from Canny no longer contains any radiometric information. In fact, the edges could be represented as a simple list of points on a Cartesian plane. The output from Canny can be used to extract higher-level geometric primitives, such as lines or corners. The edge features can also be used directly for classification and recognition algorithms, as described later in this chapter.

Radiometric and geometric methods each have different strengths and weaknesses. Compared to the radiometric methods, Canny is less sensitive to lighting changes. In the example with the white cube, the lighting changes may have little or no effect on the edges extracted. However, two cubes of entirely different color may produce nearly identical edge features, so two configurations of a Rubik's cube may appear identical. Performing recognition based on both sources of information rather than either source should provide a distinct advantage.

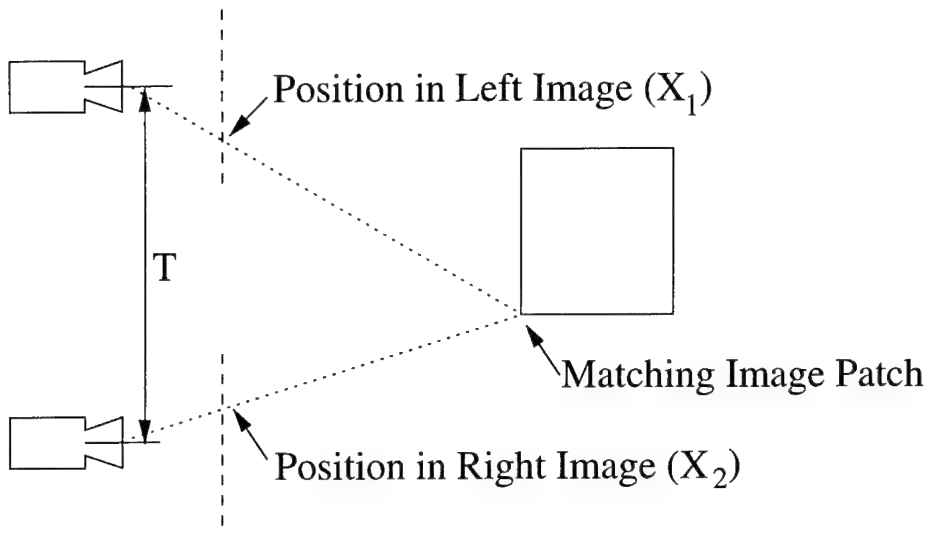

Position in Left Image (X_1)
T
Matching Image Patch
Position in Right Image (X_2)

Figure 2.3: Inferring depth information using the stereo disparity from two cameras

2.1.5 Range Versus Radiometric Features

Range data and radiometric data provide very different opportunities for an object recognition algorithm. Active range sensors, such as laser range finders, can accurately provide a wealth of purely geometric data about the scene [Bes89]. However, range data says nothing about the radiometric properties of objects, such as the illuminant response or perceived color. Radiometric data is influenced by both the surface properties of the objects in the scene and the interaction of the geometric properties of a scene with the lighting. As a result, illumination data can be used to infer both geometric and surface material properties. However, due to the interaction of scene properties, range data inferred from a radiometric image is generally less accurate than range data from an active range instrument [Bes89].

Stereo disparity infers range information from two or more images acquired from different known viewpoints [DH73, TV98]. As shown in Figure 2.3, matching image patches are found in the images and the difference of the patch location in the images indicates the patch's distance from the camera. In the simplest implementation, two cameras are used to acquire two images and establish the disparity of the image patch location. In this case, the difference or disparity is $D = X_1 - X_2$, and the distance from the camera is

$$Z = f\frac{T}{D} \tag{2.1}$$

where Z is the camera relative distance to the image patch, T is the distance between the two cameras or baseline, and f is the focal length of the cameras.

The fundamental problem with stereo disparity is quickly and accurately identifying unique image patches. Some images, such as a wall or floor with a repetitive pattern, do not contain unique image patches. Many parts of the wall in the right image match many parts of the wall in the left image. As a result, simple local stereo disparity matching may fail to estimate the range in these areas of the scene. Even when both images are uniquely patterned, it is computationally demanding to search for thousands of image patches. Pre-processing filters are usually used to estimate the sections of the images that are likely to be unique. Camera relative range limits are also used to limit the size of the image patch displacement search. The pre-processing filter, image matching errors, and range estimate limits, among other problems, limit the accuracy of stereo disparity range estimates when compared to active range sensors.

The simple feature taxonomy described above provides insight into different algorithms and indirectly suggests where complementary strengths and weaknesses may be found. For example, local methods are less affected by occlusion than global methods. Geometric methods are less affected by changing scene illumination than radiometric methods. The algorithms selected for the Extended Condensation Filter should fail under different conditions and, if each algorithm's results are fused properly, the combined estimate should be more reliable and accurate than estimates obtained using the individual methods alone.

2.2 Classifiers

Classification assigns class labels to image features. These features may be pixels, regions, or in very simple domains, the entire image. Class labels typically represent groups of known things such as corners, cubes, chairs, or cars [DH73, TV98, CA99]. However, for this dissertation, a very particular form of classification is used, where recognizing a single instance of a specific object is the goal. As a result, there are only two class labels: target object and non-target object. This form of object recognition is also called object identification [HR97]. The goal is determining if the unknown image contains an instance of the target object. For my implementation, the target object representation is a single template image of the object. Therefore, the term target image means an image known to contain a view of the object of interest.

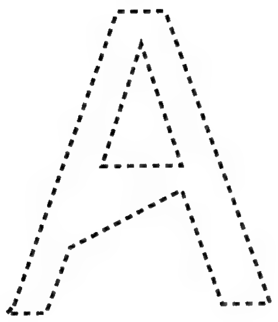


Figure 2.4: Edge features extracted by the Canny edge detector from an unknown image.

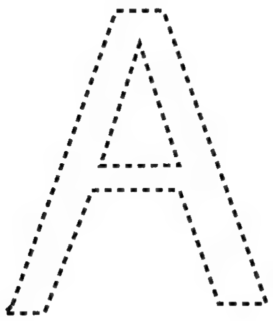


Figure 2.5: Edge features extracted by the Canny edge detector from a target image.

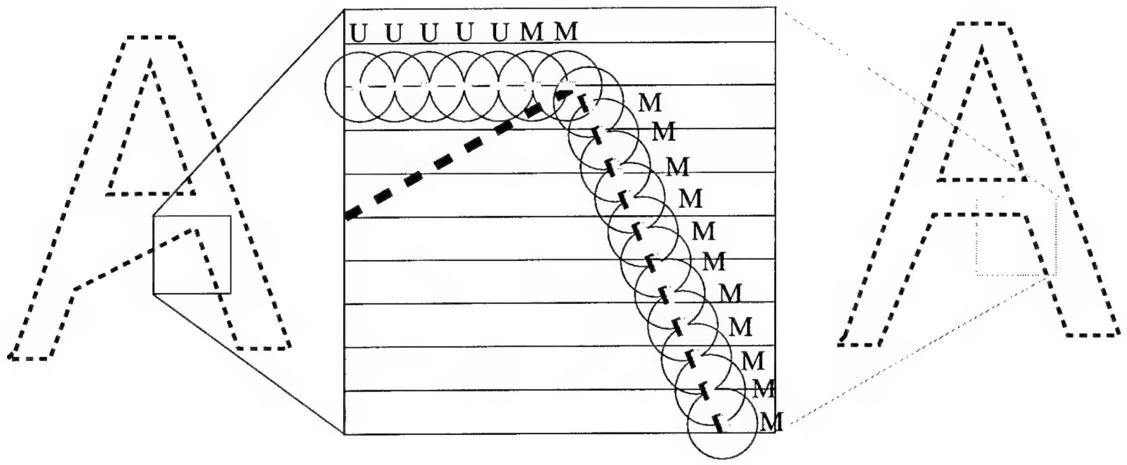


Figure 2.6: Edge feature comparison using the Hausdorff algorithm. The target edge features are shown on the right and the unknown edge features are shown on the left. The circles show the user-specified maximum distance. "M" indicates a matched edge feature and "U" indicates an unmatched features.

For example, the Canny edge detector, described in the previous section, and the Hausdorff metric can be combined to create a classification algorithm [Can86, HR92]. Assume the Canny edge detector extracted the edge features shown in Figure 2.4 from an unknown image and the edge features shown in Figure 2.5 from the target image. As mentioned earlier, each edge feature can be described as a point on a Cartesian plane. The Hausdorff distance can be used to compute the similarity between the two sets of edge features. As shown in Figure 2.6, the Hausdorff algorithm super-imposes the target and unknown edge features on the same Cartesian coordinate system. Then the algorithm tests every target edge feature by looking within a user-specified distance around the pixel for any edge pixel from the unknown image. In Figure 2.6, the circles show the user-specified maximum Euclidean distance for each target edge feature. Unmatched target edge features, where no unknown image edge feature appears within the circle, are marked with a "U" and matched edge features are marked with an "M." For this piece of the example image, the Hausdorff similarity, matched edge features divided by the total number of edge features, is 13/18 or 0.72 A variant of this algorithm, described later, will be used to test my Condensation Filter extensions.

My dissertation focuses on the single-target two-class problem. Therefore, there are only two class labels — target and non-target. If the similarity between the acquired image features and the target model exceeds the threshold, it is labeled target. Otherwise, the acquired image is labeled non-target. For example, using the Canny-Hausdorff classifier, any unknown image with a similarity score equal to or greater than 0.9 (90% matched edge features) is classified (or labeled) as an instance of the target. Any images below 0.9 will be classified as a non-target.

The Canny-Hausdorff example may lead some readers to believe my definition of classification is simply template matching. However, my definition is much broader. In general, template matching requires the comparison of pixel geometries [HS96, TV98]. My definition of classification includes a much broader class of algorithms, such as color index, which compares the color content of two images without any regard for pixel geometry [SB91]. In the case of color index, color histograms are extracted from the target image and all pixel geometry information is discarded. The unknown image and target image are compared using ratios of histogram bin values. For my purposes, classification is inclusive of algorithms like color index and much broader than template matching.

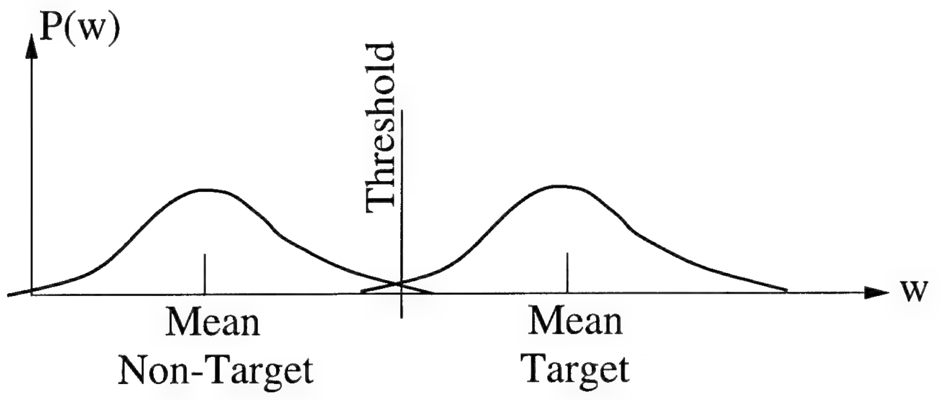


Figure 2.7: Example Gaussian distributions for a two-class classification problem. w is a similarity metric and $P(w)$ is the probability of the metric w occurring for each distribution. The threshold is used to discriminate between target and non-target images.

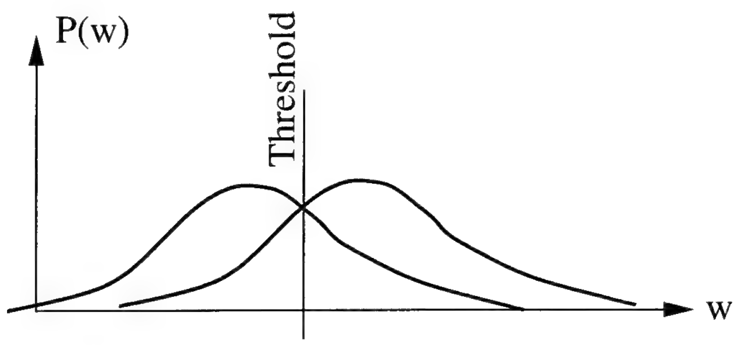
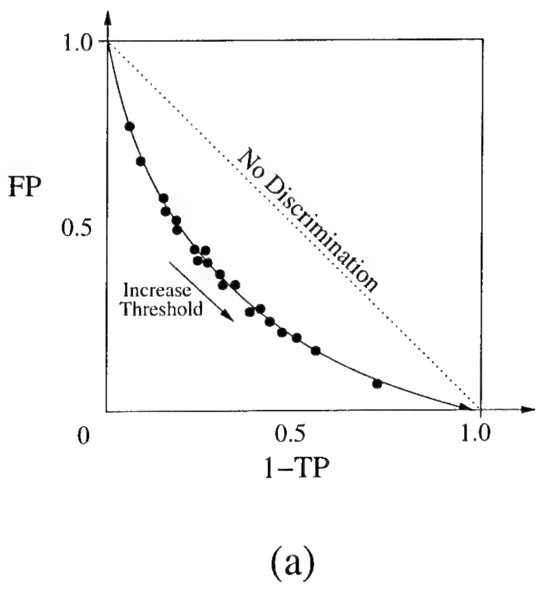


Figure 2.8: Example Gaussian distributions for a two-class classification problem where the classifier has relatively poor discrimination, when compared to Figure 2.7.

2.2.1 Discrimination

The similarity between images of the same class will vary due to a variety of factors such as changes in lighting and viewing angle. The two Gaussian distributions, shown in Figure 2.7, help illustrate this variability for the two-class problem [Swe64, Ega75]. The left Gaussian is the probability of measuring the similarity w, given an instance of the non-target, $P(w|C \neq C^{True})$. The right Gaussian is the probability of measuring w, given an instance of the target $P(w|C = C^{True})$. For this example, Figure 2.7 indicates images in the target class have greater mean similarity than images in the non-target class — a desirable result.

Ideally, the two distributions are widely separated with only a small area where the tails of the distributions overlap. If the two distributions are well-separated, a threshold can be set which segregates the two distributions. However, in the example shown in Figure



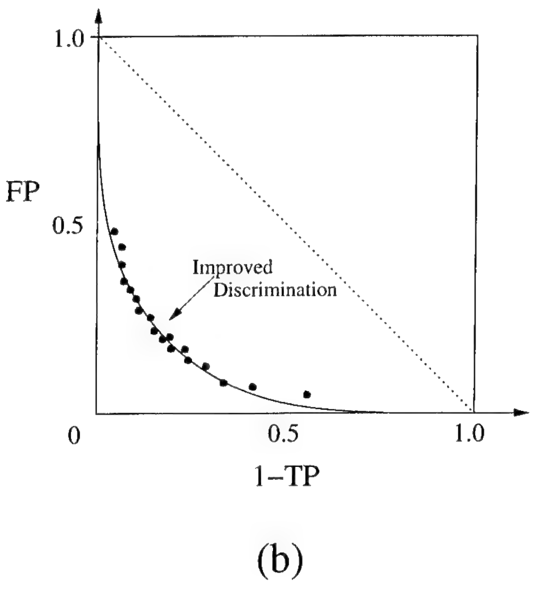


Figure 2.9: Typical ROC Curves. Figure (a) indicates less discrimination than (b).

2.7, the tails of the Gaussians are infinite, so there is no threshold value that completely segregates the two distributions — there is always a finite probability of an incorrect classification. Gaussians are used for this example, but any distribution that accurately models the probability of observing the similarity metrics is valid.

A classification algorithm with greater separation is defined as more *discriminating* than an algorithm with less separation. Figure 2.8 shows the target and non-target distributions for a classification algorithm with relatively poor discrimination when compared to the classification algorithm represented by Figure 2.7. In practice, the specific classification task will dictate the discrimination required, but the greater the separation between the two distributions, the better.

The probability of a false positive is the area in the tail of the non-target distribution $P(w|C \neq C^{True})$ above the threshold. For example, given the threshold shown in Figure 2.7, the portion of the non-target class distribution above the threshold represents the probability of incorrectly classifying a non-target image as a target image. The probability of mis-classifying a non-target image as a target can be reduced by moving the threshold to a higher value. However, increasing the threshold also decreases the probability of correctly classifying a target image as a target.

The discrimination of a classifier over a range of threshold settings can be represented by the Receiver Operating Characteristic (ROC) curve [Swe64, Ega75]. As shown in Figure 2.9, one axis of the ROC curve represents the probability that a true classification (target) exceeds the similarity threshold, also called a called a True Positive (TP) indication. The other axis represents the probability that a false classification (non-target) exceeds the similarity threshold, also called a False Positive (FP). One minus the TP probability is used to make the origin coincident with maximum discrimination and the dotted line coincident with no discrimination. As shown in Figure 2.9, the 1-TP and FP probabilities are plotted over a range of threshold settings, resulting in a sequence of points with one end of the sequence at the point $(1 - TP = 1.0, FP = 0.0)$, the result of setting the threshold at infinity, and the other end at the point $(1 - TP = 0.0, FP = 1.0)$, the result of setting the threshold at negative infinity. The points between the two extremes define the ROC curve.

The area under the ROC curve (AUC) is a single number measuring the discrimination of the classifier over a range of threshold settings. Figure 2.9(b) shows an ROC curve where the classifier is more discriminating than the classifier shown in Figure 2.9(a). The worst possible classifier would have an ROC curve that is a straight line from $(1-TP = 1.0, FP = 0.0)$ to $(1 - TP = 0.0, FP = 1.0)$. The area under the worst possible curve is 0.5 and the area under the best curve is 0.0. The AUC will be used throughout this dissertation to measure discrimination.

2.3 Combining Classifiers

Research has shown combining the results from multiple classification algorithms can create a better classification algorithm [KHDM98, JDM00]. When multiple classification systems are used together, each of the contributing classification algorithms is called an *expert* [McC81, GZ86, Hin99]. Intuitively, if all the experts agree on the class label assignment, we have greater confidence in the class label. However, if the experts disagree, or only partially agree, combining their results becomes more complex. As shown in Figure 2.10, classifier combination is a type of data fusion that combines multiple classification estimates with the goal of improving classification accuracy. Many methods have been proposed for classifier combination and, just like feature selection, a taxonomy of these methods helps explain the context and scope of this dissertation.

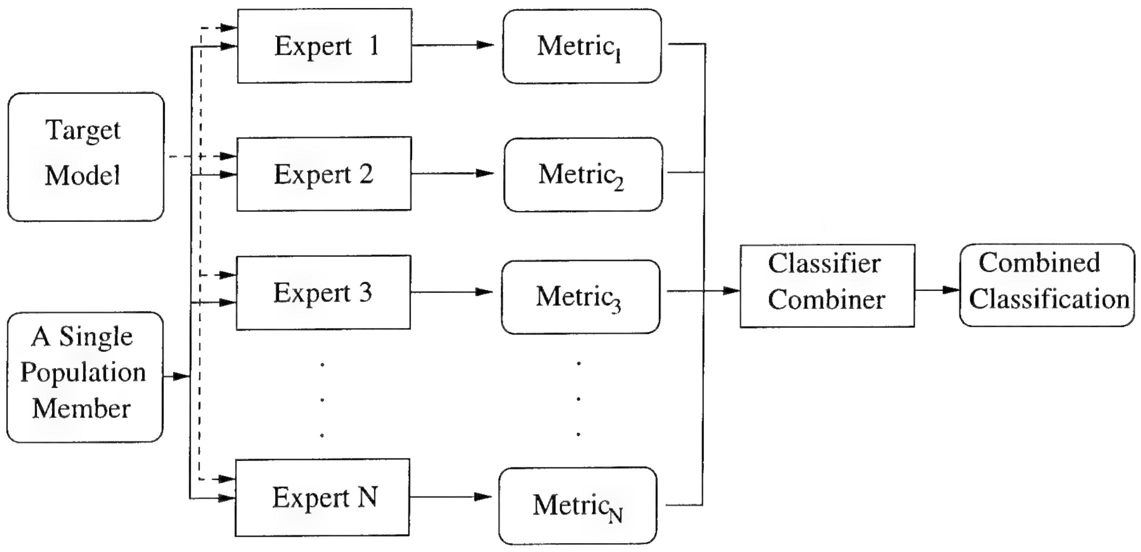


Figure 2.10: Combining classification data from multiple experts

First, classifier combination algorithms are either deterministic or probabilistic. In the mid-1980s, computer vision algorithms extracted features to create more expressive symbols, but few methods explicitly analyzed the effect of uncertainty on the feature extraction process [FB80, Kan81, Bar83]. One example is the simple voting algorithm shown in Figure 2.11. Given an unknown image, each classifier computes a similarity score. One example of a similarity score is the Hausdorff distance described earlier. For each expert, the class label with the highest similarity to the unknown image receives one vote. The class label with the largest number of votes is assigned as the class label for the unknown image. Variations of the voting algorithm allow each expert to vote more than once. For example, each expert could give three votes to the label with the greatest similarity, two votes for the label with the second greatest similarity, and one vote for the label with the third greatest similarity. However, with either variation, the form of class label assignment is crisp and definitive, information about uncertainty is lost when the classification label is assigned [RKMI95, Bor99].

Recent applications of data fusion and feature extraction to computer vision have emphasized propagating information about uncertainty throughout the computer vision process. Within the classifier combination systems that consider uncertainty, many representations and calculi are used. Pearl suggests an extensional versus intensional taxonomy [Pea88]. Extensional approaches treat uncertainty as a generalized truth value. The certainty of a

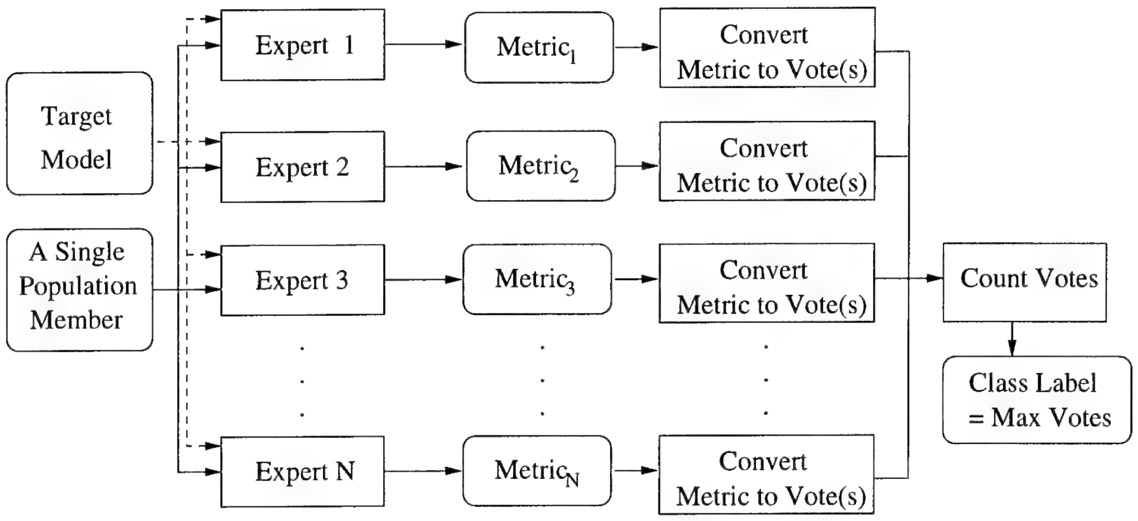
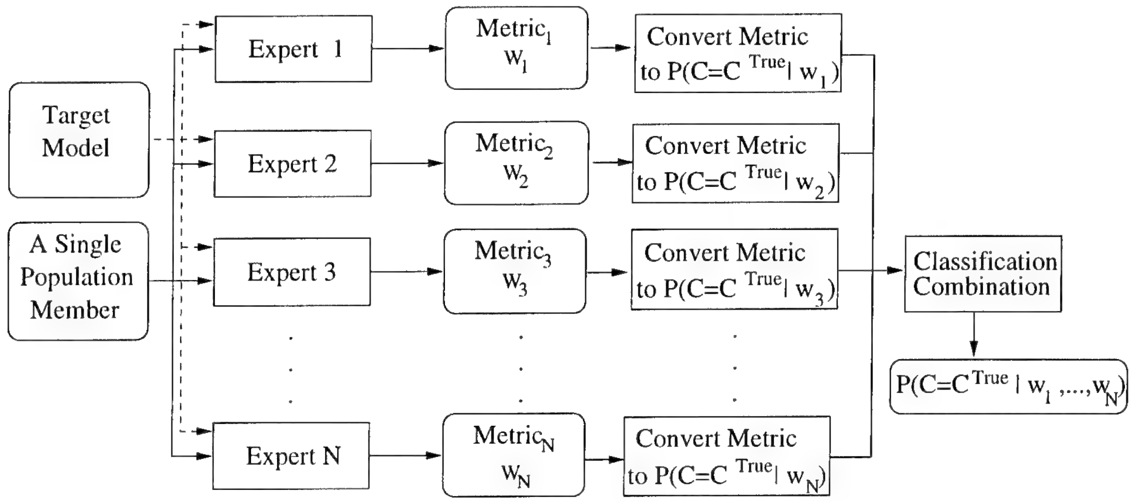

Target Model
A Single Population Member
Expert 1
Expert 2
Expert 3
Expert N
Metric$_1$
Metric$_2$
Metric$_3$
Metric$_N$
Convert Metric to Vote(s)
Convert Metric to Vote(s)
Convert Metric to Vote(s)
Convert Metric to Vote(s)
Count Votes
Class Label = Max Votes

Figure 2.11: Combining classification data using a deterministic voting algorithm and a single target model

formula is defined to be a unique function of the certainties of its sub-formulas. The extensional approaches lead to modular rule-based systems and efficient methods for inferring new information from statements. Intensional uncertainty measures are assigned to sets, and the sets are combined by set-theory operations. For example, the probability $P(A \wedge B)$ is given by the weight assigned to the intersection of the two sets A and B. Pearl argues that extensional systems are simple and efficient, but suffer from the same weaknesses as a monotonic logic, while intensional systems are far more robust but more difficult to implement [Pea88, Pea00]. The approach I propose here may be thought of as an efficient object-recognition-specific intensional system.

According to Pearl, there are four primitive relationships any probabilistic reasoning system should represent: likelihood ("The target is more likely at pose X than pose Y."), conditioning ("If the image is red, the target is at pose X."), relevance ("Whether pose X contains the target depends on whether the image is red."), and causation ("The target being at pose X caused the image to be red.") [Pea88]. Conditioning, relevance, and causation can be useful for an object recognition system, but likelihood relationships are sufficient for my purposes; the target is more likely to be at pose X than pose Y.

Determining whether one event is more likely than another translates directly into *processing time.* An object recognition system un-guided by likelihood wastes resources chasing the unlikely while neglecting the likely. Part of directing resources is also stopping the expenditure of resources when a hypothesis has sufficient support to be considered valid. The first purpose only requires a relative representation of likelihood. For example, more likely target locations should be represented by relatively larger numbers. The second purpose requires a metric suitable for comparison to a threshold.



Target Model
A Single Population Member
Expert 1
Expert 2
Expert 3
Expert N
Metric$_1$ W_1
Metric$_2$ W_2
Metric$_3$ W_3
Metric$_N$ W_N
Convert Metric to $P(C=C^{True}|w_1)$
Convert Metric to $P(C=C^{True}|w_2)$
Convert Metric to $P(C=C^{True}|w_3)$
Convert Metric to $P(C=C^{True}|w_N)$
Classification Combination
$P(C=C^{True}|w_1,...,w_N)$

Figure 2.12: Combining classification data using a probabilistic algorithm and a single target model

If operators are only required to determine relative likelihood relationships, a diverse set of approaches to classifier combination may be suitable, depending on the properties of the unclassified data and the experts. However, the probabilistic combination of the expert classifications forces a dramatic change from the simple voting system in Figure 2.11. The revised classifier combination algorithm is shown in Figure 2.12. Rather than convert each expert's distance metric into votes, a probabilistic representation requires each expert's metric be converted to the probability of the class label given the metric. Each expert i provides a metric w_i that is transformed into the conditional probability $P(C = C^{True}|w_i)$ that a specific class label estimate C is the true class label C^{True}. The function to convert w_i to a probability estimate may be learned from examples or estimated manually.

If the probability combination operator has specific properties, the uncertainty about the class label assignment is retained in the combined result $P(C = C^{True}|w_1, \ldots, w_N)$. The specific properties required for classifier combination are dependent on the properties of the individual experts. The next two sections illustrate the issues involved with combining classifiers using two simple, but widely used, examples of probability combining operators —

the product rule and the mean sum rule [KHDM98]. The product rule provides a probability estimate and assumes the experts provide independent probability estimates. The mean sum rule provides a likelihood estimate and assumes the expert probability estimates are context independent.

2.3.1 Product Rule

The product rule is derived using Bayes' Rule and the assumption of independence among the experts [McC81, GZ86, KHDM98]. When combining N experts, Bayes' Rule becomes

$$P(C = C^{True}|w_1, \ldots, w_N) = \frac{P(w_1, \ldots, w_N|C = C^{True})P(C = C^{True})}{P(w_1, \ldots, w_N)} \tag{2.2}$$

Unfortunately, some of the terms in the right side of the equation may be difficult to determine empirically. In practice, the two terms $P(w_1, \ldots, w_N|C = C^{True})$ and $P(w_1, \ldots, w_N)$ require an extremely large number of samples to estimate with confidence. To reduce the number of samples required, the independence assumption is combined with the law of conditional probability to approximate these terms. Conditional probability is defined as

$$P(A, B) = P(A|B)P(B) \tag{2.3}$$

Independence is defined as

$$P(A) = P(A|B) \tag{2.4}$$

Combining the independence assumption with the definition of conditional probability yields

$$P(A, B) = P(A)P(B) \tag{2.5}$$

Using this definition of independent conditional probability, the first term in the numerator of Equation 2.2 can be re-written as

$$P(w_1, \ldots, w_N|C = C^{True}) = \prod_{i=1}^{N} P(w_i|C = C^{True}) \tag{2.6}$$

The denominator of Equation 2.2 can be re-written as

$$P(w_1, \dots, w_N) = \prod_{i=1}^{N} P(w_1, \dots, w_N) \tag{2.7}$$

Substituting Equation 2.6 and Equation 2.7 into Equation 2.2 yields the product rule

$$P(C = C^{True} | w_1, \dots, w_N) = P(C = C^{True}) \prod_{i=1}^{N} \frac{P(w_i | C = C^{True})}{P(w_i)} \tag{2.8}$$

The independence assumption, used to simplify the estimation of the terms in Bayes' Rule, is the key assumption in the product rule. Independence simply means the probability of one event A is not influenced by the probability of another event B. This assumption is a key component of results provided later in this dissertation.

2.3.2 Product Rule Example

Table 2.1: Table of Objects in the Simple Product Rule Example World

Object (symbol)	Color	Height
Stop Sign (S)	Red	Med
Stop Sign (S)	Red	Med
Fire Hydrant (F)	Red	Low
Bird (B)	Green	Low
Tree (T)	Green	High

Although independence among probability estimates is a common assumption, it is easy to create an example where the independence assumption leads to errors. My example uses the simple world shown in Table 2.1. The world has only five objects and four class labels; stop sign, fire hydrant, bird, and tree. In my simple world, all of these object classifications have only two identifying features; color and height. All colors in the world are either red or green and all heights are either low, medium, or high.

Given color or height observations, the probability of an object's class label can be calculated. For example, given the object is of low height (L), Bayes' Rule can be used to calculate the probability it is in the class fire hydrant (F)

$$P(C = F | L) = \frac{P(L | C = F) P(C = F)}{P(L)} = \frac{1.0 * 0.2}{0.4} = 0.5 \tag{2.9}$$

Table 2.2: Testing the independence assumption in the Product Rule example. If independence were a valid assumption, $P(A) = P(A|B)$.

$P(A)$	$P(A\|B)$
$P(R) = 0.6$	$P(R\|L) = 0.5$
$P(L) = 0.4$	$P(L\|R) = 0.33$

This result can be verified directly from the table, by observing that a low height object can only be either a fire hydrant or a bird and the $P(C = F|L) = 0.5$

Given an object is red (R), the probability of it being in the class fire hydrant is

$$P(C = F|R) = \frac{P(R|C = F)P(C = F)}{P(R)} = \frac{1.0 * 0.2}{0.6} = 0.33 \tag{2.10}$$

This result can also be verified by examination of the table.

Given both color and height observations, the probability of a class label may be calculated with greater certainty. Given the object is both low height and red, the certainty of my class label assignment is much higher. Using the Bayesian Product Rule, the probability of a low-height and red object being a fire hydrant is

$$P(C = F|L, R) = P(C = F)\frac{P(L|C = F)P(R|C = F)}{P(L)P(R)} = 0.2\,\frac{1.0 * 1.0}{0.4 * 0.6} = 0.83$$

However, this result does not agree with the results from direct examination of the table. From the table $P(C = F|L, R) = 1.0$, not 0.83. The independence assumption is the source of the discrepancy. Looking back at the derivation of the Bayesian Product Rule, the denominator $P(L)P(R)$ is a substitution for the $P(L, R)$ based on the independence assumption. The product $P(L)P(R)$ is equal to $0.4*0.6$, but from inspection $P(L, R)$ is 0.2. Table 2.2 indicates independence is frequently a bad assumption in my example problem domain. The independence assumption will become important again later during testing in a much more complex object recognition problem domain.

2.3.3 Normalization

In Equation 2.8, the conditional probability of a class label is derived from estimates of $P(C = C^{True})$, $P(w_i|C = C^{True})$, and $P(w_i)$. For most applications, each of these three probabilities is estimated and includes some error. The denominator of equation 2.8 also includes the independence assumption, which the earlier example shows may be incorrect. To the degree the estimates and independence assumption are incorrect, the probability of all possible outcomes may not equal 1.0 and, as a result, the probability estimates require normalization.

The effect of the error produced by the independence assumption can be seen in the previous example. As shown earlier, using the product rule, the probability that an object is in the class fire hydrant, given the identifying features red and low height is $P(C = F|L, R) = 0.83$. Examining the table shows the probability of any other identifying features, given the class fire hydrant, is 0. For example, given the observation green, the probability of a fire hydrant is

$$P(C = F|L, G) = P(C = F)\frac{P(L|C = F)P(G|C = F)}{P(L)P(G)} = 0.2\,\frac{1.0 * 0.0}{0.4 * 0.4} = 0.0$$

and given the observation medium height, the probability of a fire hydrant is

$$P(C = F|M, R) = P(C = F)\frac{P(M|C = F)P(R|C = F)}{P(M)P(R)} = 0.2\,\frac{0.0 * 1.0}{0.4 * 0.6} = 0.0$$

Therefore, the probability of all possible class labels is

$$P(C = S|L, R) + P(C = F|L, R) + P(C = B|L, R) + P(C = T|L, R) = 0.83$$

However, this result cannot be correct. By definition, the probability of an object (from our closed world) being in one of the four classes is 1 — the object must be something. Therefore

$$P(C = S|L, R) + P(C = F|L, R) + P(C = B|L, R) + P(C = T|L, R) = 1 \qquad (2.11)$$

The incongruous results are due to the independence assumption in my simple example. In more complex applications, such results may also be caused by estimation error.

Since the sum of the probability of all possible outcomes must equal 1, a constant α can be derived to normalize the probabilities, where

$$\alpha = \frac{1}{P(C = S|L, R) + P(C = F|L, R) + P(C = B|L, R) + P(C = T|L, R)} \qquad (2.12)$$

To normalize the relative likelihood values, α is used as follows

$$P'(C = F|L, R) = \alpha P(C = F|L, R) \qquad (2.13)$$

Where P' is the probability estimate after normalization. In the example, the constant

$$\alpha = \frac{1}{0.0 + 0.83 + 0.0 + 0.0}$$

The class label F given L, R has a probability 1

$$P'(C = F|L, R) = \frac{1}{0.83}0.83$$

and the other class labels have a probability of 0.

Normalization provides probabilities that sum to 1.0 at the expense of computing the probability estimate for all the class labels and normalizing the sum of the values to 1. For the remainder of this dissertation the notation $P(\ldots)$ indicates a normalized probability and the notation $\hat{P}(\ldots)$ indicates a normalized probability estimate.

2.3.4 Sum Rule

The weighted sum of expert opinions is an intuitively appealing method for combining multiple classification estimates [McC81, GZ86, KHDM98, Hin99]. Given the classification metrics $(w_1, \ldots, w_N)$ from N experts, the probability estimate using the Sum Rule is the weighted sum of the expert estimates

$$L(C = C^{True}|w_1, \ldots, w_N) = \sum_{i=0}^{N} \beta_i P(C = C^{True}|w_i) \tag{2.14}$$

where β_i is the weight given to each expert's probability estimate. To preserve a normalized probability estimate, the weights must sum to 1.0.

If each expert's estimate is given equal weight, the probability estimate becomes the mean value of the expert estimates, called the Mean Sum Rule

$$L(C = C^{True}|w_1, \ldots, w_N) = \frac{\sum_{i=1}^{N} P(C = C^{True}|w_i)}{N} \tag{2.15}$$

This combination rule is intuitively appealing. If multiple experts estimate a probability and they make independent and unbiased random errors, in accordance with the central limit theorem, the mean value of their estimates should approach the true value as the number of experts approaches infinity.

The idea of combining multiple experts with independent and unbiased experts in the attempt to eliminate the bias is also a concept supporting the bagging algorithm [Bre94].

However, there is and important difference between my pre-defined experts and bagging. The experts contributing to the Extended Condensation Filter, such as the Hausdorff algorithm or color index, are pre-defined — the nature of their biases is unknown. In comparison, the bagging algorithm is careful to select training sets which should create random biases in the contributing experts — Breiman calls these unstable procedures [Bre94]. However, as results later in this dissertation show, the experts chosen for my implementation may be sufficiently independent and unbiased to provide the benefits of bagging.

Kittler shows the Mean Sum Rule can be derived using Bayes' Rule [KHDM98]. To give each expert's estimate equal weight, context independence must be assumed. In other words, none of the experts has any information or expertise unavailable to the other experts. The context independence assumption is justified under some circumstances. To illustrate, if ten stock market experts are asked the probability a market index will go up tomorrow, it is reasonable to assume each is working within the same context. In contrast, the earlier fire hydrant example differs from the stock market example. In the fire hydrant example, the two "experts" observe different features: color and height.

In addition to context independence, Kittler's derivation requires the conditional probabilities computed by the respective classifiers not deviate dramatically from the prior probabilities. Essentially, Kittler's derivation assumes the experts provide little, if any, information. Kittler admits this is a "rather strong assumption", but argues it "may be readily satisfied when the available observational discriminatory information is highly ambiguous due to high levels of noise." Despite the simple intuitive appeal of averaging expert estimates, Kittler's derivation from Bayesian principles relies on assumptions with questionable general applicability.

2.3.5 Sum Rule Example

The Mean Sum Rule may be intuitively appealing, but it does not work well for the earlier fire hydrant example. From Equation 2.9, if the object is observed to be of low height, the probability of the class label fire hydrant is $P(C = F|L) = 0.5$. From Equation 2.10, if the object is red, the probability of the class label fire hydrant is $P(C = F|R) = 0.33$. Using the Mean Sum Rule to combine these estimates yields

$$L(C = F|L, R) = \frac{P(C = F|L) + P(C = F|R)}{2} = \frac{0.5 + 0.33}{2} = 0.4$$

However, as noted in the Bayesian Product Rule example, the actual probability, based on examination of the table, is 1.0. The Mean Sum Rule estimate is further from the true 1.0 value in the table than the product rule estimate of 0.83.

The context independence assumption is the source of the discrepancy because the two experts are looking at completely different aspects of the problem. One looks at color only. The other looks at height only. Each expert responds differently depending on the class label and observation. For an extreme example, consider the tree class label. When the height expert observes a high object, it estimates $P(C = T|H) = 1.0$, while the color expert observes green and estimates $P(C = T|G) = 0.5$. When the experts are context dependent, large estimation errors can be expected. The Mean Sum Rule will be tested again in a far more complex object recognition problem domain later in this dissertation.

2.3.6 Non-Uniform Weights

Context independence is frequently a bad assumption because the expert opinions may not be equally discriminating. If we discover that one expert is consistently more discriminating, we should give that expert's estimate greater weight. However, classifier discrimination may be a function of the conditions when the estimate was made. In other words, the weight for each expert needs to be a function of the state of the classification problem. For example, one expert may provide more discriminating estimates in close proximity to the camera, another expert may be more discriminating under bright lights. This condition dependence is expected, considering the feature taxonomy described in the first section of this chapter. To change the weight under specific conditions, the weight can be represented as a function $\beta_i(conditions)$, where the term *conditions* is an entirely problem dependent set of conditions such as lighting or camera relative distance to the target estimate.

$$L(C = C^{True}|w_1, \ldots, w_N) = \sum_{i=0}^{N} \beta_i(conditions) P(C = C^{True}|w_i) \tag{2.16}$$

Weighting functions can be used to compute each expert's weight as a function of conditions that affect the expert's reliability. The weights could simply be estimated by a human expert, or autonomously learned by a machine learning algorithm, such as a neural network or boosting.

In addition to learning a set of non-uniform weights, boosting directs the training process [FS99]. Boosting algorithms, such as the AdaBoost algorithm described in [FS99], focus each expert's training on the cases most often mis-classified by that expert. For my application, boosting would focus training on those target or non-target poses which are incorrectly classified.

Rather than add the complications of a neural network or boosting algorithm, a very simple training process and combination rules were implemented for this dissertation. However, nothing encountered during this research indicates the Extended Condensation Filter cannot take advantage of the more sophisticated algorithms, such as neural networks, bagging, or boosting. Demonstrating the compatibility of the Extended Condensation Filter with these more sophisticated methods remains an open research question. The training process and combination rules implemented for this dissertation will be discussed during the description of the Extended Condensation Filter implementation in 4.

2.4 Estimating Location

The classification process discussed in the previous section is one of the two components of object recognition — localization is the other [TV98]. Classification measures the similarity of the target image and an unknown image. Recognition assumes multiple potential targets may be visible in the unknown image. Rather than compute a similarity metric between the target image and the entire unknown image, a recognition algorithm must compare portions of the unknown image with the target image to localize potential targets. The complete pose estimate for any non-deformable object requires three translation parameters and three rotation parameters — the standard six-degrees-of-freedom for any non-deformable object. If the camera location in the world is known, these six camera-relative parameters can be transformed to world-relative parameters.

Given any pose estimate, a classification algorithm can be used to measure the similarity between the target image and the portion of the unknown image represented by the pose. To make this comparison, either the target image must be transformed to the unknown image based on the pose estimate, as shown in Figure 1.3 on page 5; or the portion of the unknown image representing the pose estimate must be transformed to the coordinates of the target, as shown in Figure 1.2 on page 4. Throughout this dissertation I use the target to unknown image transformation because previous research has shown reducing the

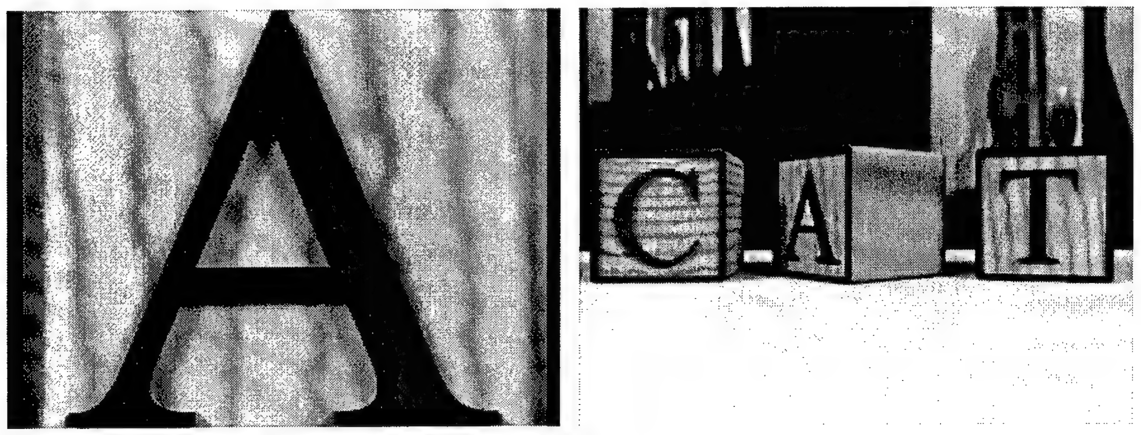


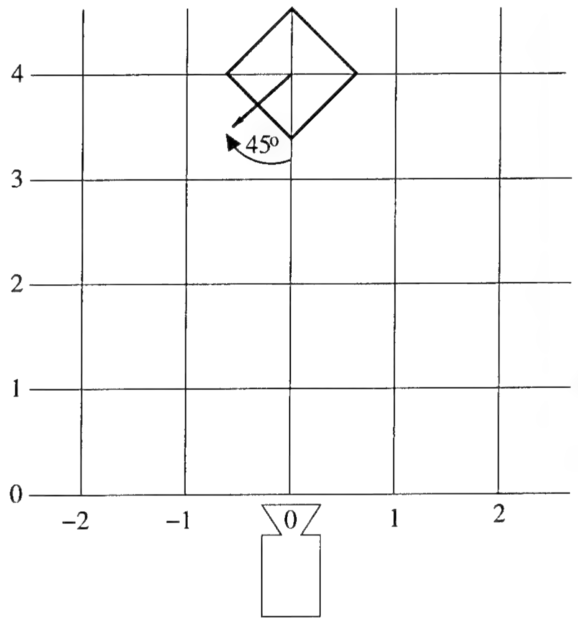
Figure 2.13: A single image that serves as the target model and an instance of the target in an acquired image

resolution of the larger target image is less error prone than increasing the resolution of a smaller unknown image [SB01].

To illustrate the target to image approach, assume the image shown on the left of Figure 2.13 is the target model and the image shown on the right is the unknown image. To simplify the example, I have reduced the problem from six-dimensions to three-dimensions by assuming the target object can only translate and rotate on the ground plane. With this simplification, only the three parameters; left-to-right translation x, translation into the image z, and ground-plane rotation θ remain. Note, the same three-dimensional coordinate system will be used later when testing the Extended Condensation Filter.

This simplification from six to three dimensions is not a minor point. Increasing the dimensionality of a problem can have unforeseen consequences. However, to make the results of this first look at the Extended Condensation Filter more manageable in terms of training time, the volume of data, and interpretation of data, the simplification is appropriate. The three dimension assumption, or ground plane restriction, is also commonly found in object recognition problems.

We want to test the similarity of the target model transformed to the pose x=0, z=4, $\theta = -45°$ shown in Figure 2.14, with the unknown image. Using the target to image approach, my question becomes, "If I put the target image at the point in the image corresponding to the pose 0, 4, $-45°$, how similar is it to what I see in the unknown image?" Figure 2.15 shows the affine transformation of the target image using the pose estimate



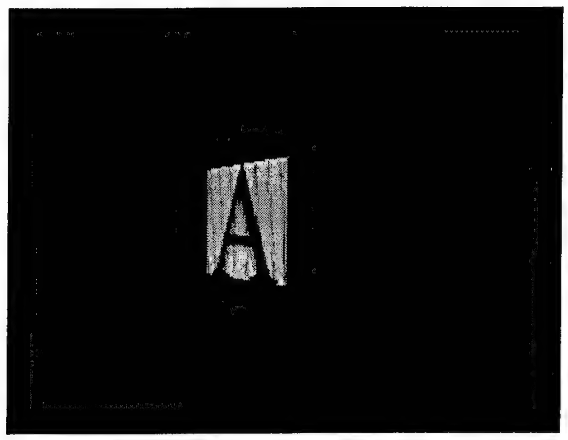


Figure 2.15: Transformed target image based on the pose shown in Figure 2.14

Figure 2.14: The camera relative pose $[x, z, \theta]$ or (ground-plane translation, depth translation, ground-plane rotation), where $[x = 0, z = 4, \theta = -45°]$.

0, 4, −45°. Every pixel in the target image was multiplied by an affine transformation matrix representing the camera relative coordinates 4, 0, −45°. Only the pixels from the transformed target image, not the black background pixels, and the exact same pixels in the unknown image are used to compute the similarity metric. This transform and compare approach is the basis for the Render-Match-Refine algorithm developed in [SB01]. There are other ways to perform object localization [Bal81, Bev93], but this approach of turning classifiers into recognition algorithms by searching in pose space is well suited to extending the Condensation Filter.

2.4.1 Evaluating Recognition Algorithms

Object recognition algorithms perform classification and localization. The classification performed by an object recognition algorithm can be evaluated like a classification algorithm using the AUC described earlier. The localization provided by an object recognition algorithm can be evaluated by two orthogonal metrics — precision and accuracy. Precision measures the dispersion of the pose estimates. Accuracy measures the proximity of pose estimates to the ground truth. Figure 2.16 shows a precise, but relatively inaccurate set of location estimates. Figure 2.17 show a relatively accurate, but imprecise set of location estimates.

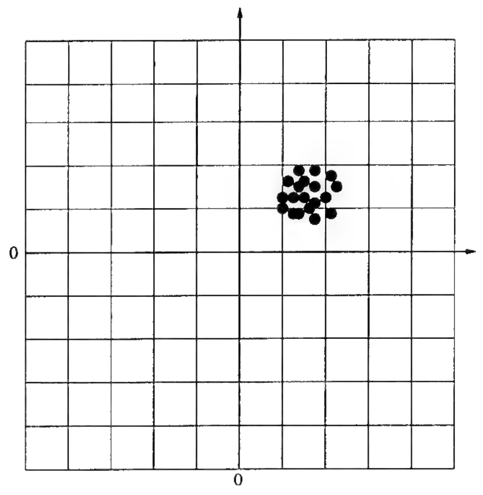


Figure 2.16: Relatively precise, but inaccurate. The estimates are tightly grouped but the mean estimate is far from the origin when compared to Figure 2.17

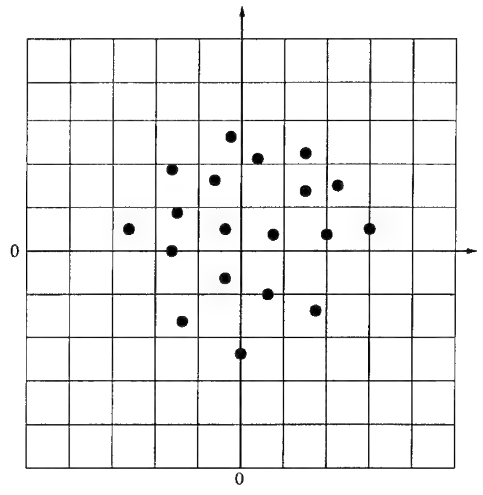


Figure 2.17: Relatively accurate, but imprecise. The mean estimate error is near zero but the estimates are not tightly grouped.

If two object recognition algorithms provide Gaussian one-dimensional pose estimates, it is easy to compare the precision and accuracy of the two algorithms. The standard deviation of the error measures precision, and the mean error measures accuracy. Unfortunately, object recognition algorithms provide multi-dimensional, typically one to six dimensional, pose estimates.

Evaluating either precision or accuracy by directly comparing multiple dimensions is difficult. For example, if an object recognition algorithm uses six dimensions and is 10% more accurate in four dimensions, but 20% less accurate in the other two dimensions, which algorithm is more accurate? Comparing the accuracy and precision in both rotation and translation also requires some form of standardization because pose estimate generally include translation and rotation — how should one unit of rotation error be compared to one unit of translation error? The circular error probable (CEP) and scaled CEP statistics, which are introduced in Chapter 4, measure accuracy and precision while accommodating these issues.

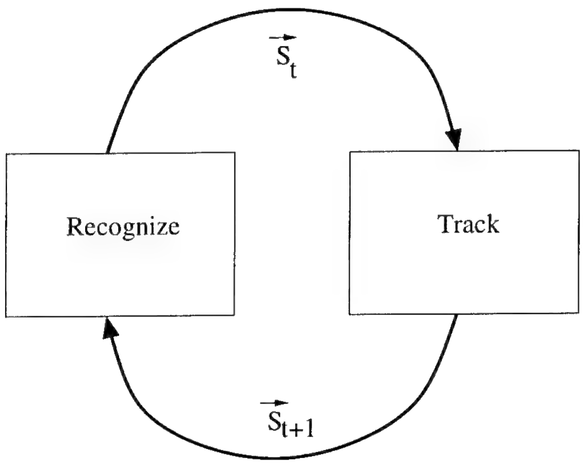


Figure 2.18: Recognition and tracking as an iterative cycle

2.5 Combining Location Estimates

Algorithms that combine location estimates have been the subject of research for over 50 years. Research on recognizing and tracking airborne targets initially drew interest with the development of radar and long range aircraft[Chu48]. The British were especially interested in using multiple radars to discriminate between aircraft and atmospheric phenomena, localizing the aircraft, then tracking the aircraft over time. Each of the three steps was essential to determining where to send intercepting aircraft. Classification, recognition, and tracking continue to have this complementary relationship.

2.5.1 Tracking

Unless one assumes a completely static environment, object recognition is enhanced by performing some form of tracking. Object recognition estimates the current pose of the target object, while tracking estimates the next pose of the target object. When object recognition is successful, it accurately and precisely estimates the current pose of the target object. When tracking is successful, it accurately and precisely predicts the next pose of the target object.

In a dynamic environment, recognition and tracking work in a complementary manner — recognition and tracking can be viewed as two steps in an iterative cycle. As shown in Figure 2.18, the recognition algorithm provides estimates of the current pose, a multi-dimensional vector $\vec{S}_t$, to the tracking algorithm and the tracking algorithm gives estimates of the pose during the next iteration $\vec{S}_{t+1}$ back to the recognition algorithm. In military terminology, successful object recognition is called acquisition of the target and successful acquisition of the target near the tracking algorithm prediction is called tracking the target.[1]

Tracking can also benefit from multiple experts, just like classification. Each expert provides an estimate of the target pose at the current iteration and, using a method dependent on the specific tracking algorithm, these estimates are combined to create a single best-estimate of the next object pose. The next three sections discuss three algorithms which have their roots in object tracking and the combination of multiple object pose estimates. Each algorithm is interesting because it offers a ready-made method for tracking objects and combining object pose estimates.

2.5.2 Kalman Filter

The Kalman Filter has a strong theoretical foundation and a long history of application [Kal60, Sor70, May79]. It is robust and accurate for systems where each sensor suffers only from unbiased Gaussian noise, target motion can be described as a vector of real values, and the next target pose is a linear function of the previous target pose and the current target pose[Kal60, May79]. However, the Kalman Filter works poorly in clutter which causes the conditional probability estimate of the target position to be multi-modal and non-Gaussian [GSS93, IB98]. More recent filters use more flexible methods capable of representing multi-modal and non-Gaussian distributions.

2.5.3 Mixture of Gaussians Filter

The Mixture of Gaussians Filter makes the weaker assumption that the underlying probability distribution is an *ad hoc* combination of Gaussian distributions [Pao94]. The multiple hypothesis approach is a more flexible way of modeling uncertainty. The conditional probabilities of a number of possible associations are evaluated. For example, a new measurement can be associated with a previous distribution, accepted as a new distribution, or rejected as clutter. Rather than hold all possible measurements as distributions in the mixture, a gating technique is applied to eliminate very low probability measurements. For many

[1]This terminology is based on 11 years of experience evaluating aircraft at the Air Force Flight Test Center, evaluating ballistic missiles at the United States Western Range, and reviewing United States Air Force doctrine.

applications, a large number of hypotheses may remain after gating, and it is often desirable to perform further mixture reduction after the measurement update [GSS93, Pao94]. Mixture reduction attempts to combine some of the Gaussians in the mixture by looking at the similarity of their statistical properties. The Mixture of Gaussians Filter is limited to systems where each sensor suffers from unbiased Gaussian noise.

Readers familiar with Expectation Maximization may recognize the Mixture of Gaussians' problem as a maximum-likelihood problem. Given a set of N observations $X = \{x_1, \ldots, x_N\}$ and the assumption that the underlying distribution $p(x|\Theta)$ is composed of a set of Gaussians, Θ represents the set of means and covariances for these Gaussians. The goal is a set of parameters Θ that best explain the observations X [Bil97]. The goal is finding the values of Θ that maximize the likelihood equation

$$L(\Theta|X) = \prod_{i=1}^{N} p(x_i|\Theta)$$

Expectation Maximization has been shown to find the optimal parameter set Θ. However, the Expectation Maximization algorithm requires the number of Gaussians be specified as an input [Bil97, DLR77]. Like Expectation Maximization, the goal of the Mixture of Gaussians filter is determining the parameters for a set of Gaussians that best explain the observations. However, the Mixture of Gaussians Filter must also estimate the number of Gaussians in the mixture. When the Mixture of Gaussians Filter is applied to an object recognition problem, assuming a specific number of Gaussians would imply the exact number of target and false target sources is known — this is usually not the case. This inability of the Mixture of Gaussians Filter to assume a specific number of Gaussians leads to the mixture reduction problem — what is the optimal number of Gaussians in the mixture?

2.5.4 Condensation Filter

The Condensation Filter avoids the limitations of the Kalman Filter and Mixture of Gaussians Filter by using a more direct approach. According to Gordon and Blake, the Condensation Filter uses weighted samples (or particles) to represent the probability that a given pose is the true target object pose conditioned on the expert observations [GSS93, IB98]. Each particle is composed of a pose $\vec{S}$ and a probability estimate $\hat{P}(\vec{S} = \vec{S}^{True})$. Using particles to represent the conditional probability has many advantages. The calculations associated with each particle are simple and fast. By using samples to represent the con-

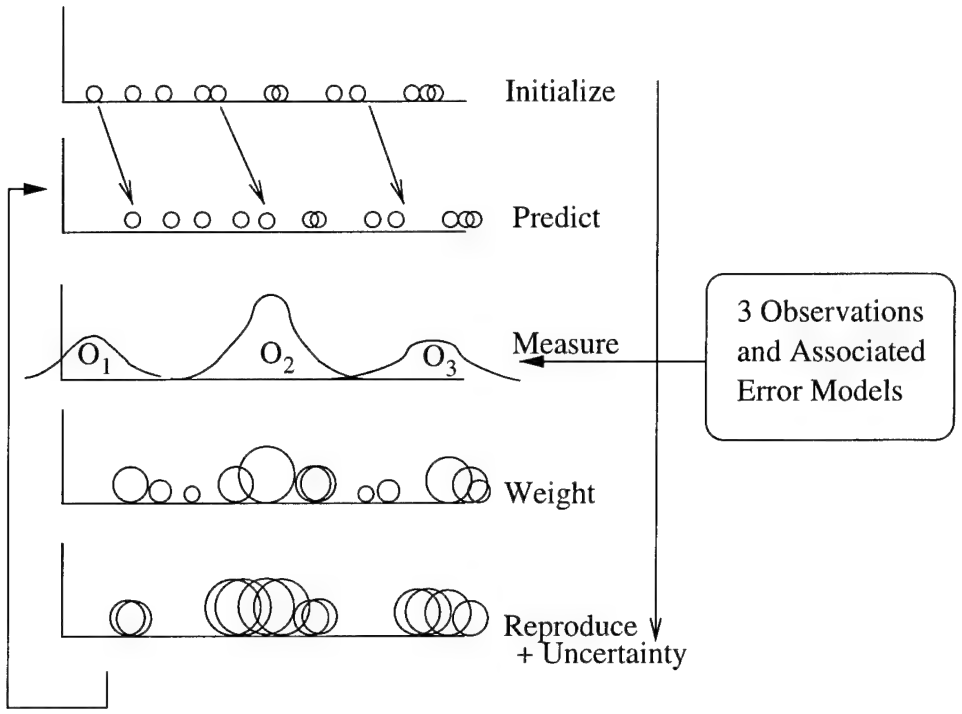


Figure 2.19: Condensation Filter process steps: 1) a filter update begins with an equally weighted set of particles, 2) the predicted motion model and a model of uncertainty are applied to every particle, 3) measurements are acquired, 4) each particle is weighted as a function of the distance between the particle and the measurements, 5) the particles are reproduced as a function of their weight

ditional probability of the target location, association hypotheses from previous time steps are not required for conditional probability evaluation[IB98].

Comparing the Mixture of Gaussians Filter to the Condensation Filter shows the importance of this last point. For each new measurement in a Mixture of Gaussians Filter, the algorithm must decide if the measurement belongs to one of the existing Gaussians, a new Gaussian, or is a noise measurement. The association problem can result in an exponential growth of Gaussians in the mixture. A mixture reduction step is required to identify and combine Gaussians that are probably from the same event [GSS93, Pao94]. This step is not required for a Condensation Filter.

Figure 2.19 illustrates the Condensation Filter update cycle. The Condensation Filter particles are shown as circles at the top of the Figure and the diameter of the circle represents the particle's weight. The filter is typically initialized with particles distributed uniformly throughout the measurement space. As measurements are received, the weight for each particle is set proportional to the distance between the particle and the measurement —

the measure step in the figure. After the particle weights are set, particles are duplicated in the next generation with a probability proportional to their weight — the reproduce step. As a result, particles that were closer to a measurement are more likely to be reproduced. If a model is available to predict the target motion, each particle is moved to anticipate the target object's next position. Random particle motion, typically a Gaussian random variable, may also be added at the weighting or prediction steps to represent uncertainty. The entire process repeats when the next set of measurements become available. A more detailed example incorporating some of my extensions will be described in the next chapter.

2.5.5 Applications of the Condensation Filter

The strength of the Condensation Filter is indicated by the large number of projects successfully using the filter. This section will cover only a few of the better known examples in computer vision.

Gordon introduced the Bootstrap Filter in 1993 [GSS93]. The Bootstrap Filter uses a population of weighted particles to represent the underlying conditional probability associated with some process — it is fast and flexible, but it is not provably optimal like the Kalman Filter.

Blake's work represents one of the first applications of the Bootstrap Filter to computer vision [IB96]. Using the descriptive name *Condensation Filter*, his application tracked objects using the parametric representation of occlusion boundaries [IB96]. In this problem domain the Condensation Filter estimates the parameters of a geometric primitive, such as a Bezier curve, that conform to the occlusion boundary of the target object [Bla92]. The occlusion boundaries are very useful for estimating the target object pose. Since the original work by Blake, other projects have applied the Condensation Filter in a similar manner [IB98, Mac00].

A recent application of the Condensation Filter can be found at the Smithsonian giving tours. The tour guide robot, named MINERVA, estimates its two-dimensional pose on the floor of the museum using a Condensation Filter [TFB98]. MINERVA compares images of the ceiling to its internal database of ceiling images. The most similar image indicates the robot's location. Unfortunately, many ceiling images are not unique and a single image may only narrow the range of possible robot locations rather than indicate a unique location. Acquiring additional images eventually reduces the number of possible locations. A

Condensation Filter is used to maintain multiple hypotheses initially and as more ceiling images are acquired, a single two-dimensional pose estimate is supported. A presentation on MINERVA at the IEEE Computer Vision and Pattern Recognition conference was my first exposure to the Condensation Filter [D$^+$99].

The Kalman Filter, Condensation Filter, and hybrids such as the Mixture of Gaussians Filter each have their strengths and weaknesses. The Kalman Filter has the strongest theoretical foundation and a long history of application in automated navigation and target tracking. However, the Kalman Filter's assumptions are very restrictive. The Mixture of Gaussians Filter is similar to the Kalman Filter; so it gains some of the theoretical support, but it also retains some of the Kalman Filter's assumptions about the process distribution. The Condensation Filter has less history and a weaker theoretical foundation, but is far more flexible than either the Kalman Filter or Mixture of Gaussians Filter. The theoretical underpinnings of the Condensation Filter will be addressed in the next chapter.

2.6 Evolution Strategies

The Condensation Filter is typically viewed as a statistical filter [IB96, IB98, D$^+$99], where expert measurements are expressed as poses, the distance between measurements and the particle pose is used to determine the particle pose probabilities, and the density of particles per unit of pose space represents the conditional probability that a portion of the pose space is the true target pose. However, the Condensation Filter is also similar to an Evolution Strategy [DL00]. Evolution Strategies are a class of search algorithms loosely based on the mechanics of natural selection. Fogel describes the Evolution Strategy as shown on the left side of Table 2.3 [Fog00]. The right side of Table 2.3 shows the comparable step for the Condensation Filter.

In an Evolution Strategy, the search for an optimal solution is performed on many points in the search space simultaneously, unlike most traditional hill-climbing techniques that work on a single point at a time. Each population member contains a candidate solution and a measure of how well it fits the ideal solution. This fitness value guides the selection process, which favors highly fit individuals, to produce new members for the next population. This allows the mixing of parental information with the goal of making the next generation more fit. The Gaussian random variable is introduced to explore portions

Table 2.3: Comparing the Evolution Strategy and the Condensation Filter

Evolution Strategy	Condensation Filter
The problem is defined as finding the real valued n-dimensional vector $\vec{x}$ that is associated with the minimum or maximum of a function $F(\vec{x})$.	Object recognition is defined as finding the real valued n-dimensional vector $\vec{S}$ estimated to represent the most probable pose of the target object $P(\vec{S} = \vec{S}_{True})$.
An initial population of parent vectors, $\vec{x}_0, \ldots, \vec{x}_N$, is selected at random from a feasible range in each dimension of the state space. The distribution of initial trials is typically uniform.	An initial population of particles containing the pose vectors, $\vec{S}_0, \ldots, \vec{S}_N$, is selected at random from the entire pose space. The distribution of initial particles poses is typically uniform.
An offspring vector, $\vec{x}'_i$ is created from each parent $\vec{x}_0, \ldots, \vec{x}_N$ by adding a Gaussian random variable with zero mean and pre-selected standard deviation to each component of $\vec{x}$.	An offspring particle containing the pose $\vec{S}'_i$ is created from each parent particle's pose $\vec{S}_0, \ldots, \vec{S}_N$ by adding a random variable representing the predicted target object motion — usually a Gaussian random variable with non-zero mean.
A selection function, such as fitness proportional, is used to determine which of these vectors to maintain in the fixed-size population by ranking the errors $F(\vec{x}_i)$ and $F(\vec{x}'_i)$. The vectors that possess the least error become the new parents for the next generation.	A selection function, such as fitness proportional, is used to determines which of the new particles to maintain in the fixed-size population by ranking the errors $F(\vec{S})$. The typical Condensation Filter does not examine both the old $F(\vec{S})$ and new $F(\vec{S}')$ generation for fitness.
The process of generating new trials and selecting those with the least error continues until a sufficient solution is reached or the available computation is exhausted.	The process of generating new trials and selecting those with the least error continues until a probable target is identified or the available computation is exhausted.

of the search space impossible to explore through selection alone. This allows innovation in the population and reduces the probability the population will be trapped in local optima.

Similar to an Evolution Strategy, the Condensation Filter works on many points in the pose space simultaneously. Each particle is a member of a fixed-size population used to produce a new population based on a fitness function. Each particle contains a candidate solution — a pose. Each particle also contains a fitness value — an estimate of the probability that the pose is the target object. The fitness values guide the particle selection process used to create the next filter population. The filter's stochastic motion models force innovation in the population and the exploration of novel portions of the search space.

As shown in Table 2.3, there are two distinct differences between the Evolution Strategy and the Condensation Filter. First, the Evolution Strategies fitness function is not necessarily a conditional probability. Second, the Evolution Strategy does not select the fittest members from both the current generation and the next generation. Despite these differences, the two algorithms contain many similarities and these similarities will be used to explore more fully what the state of the Condensation Filter particles represents in the next chapter. The key point is the Condensation Filter appears to be a form of population-based search. DelMoral provides a more rigorous mathematical treatment of the similarities between these algorithms [DL00].

2.7 Summary

This chapter has quickly reviewed a large amount of material. Several observations are worth repeating and emphasizing because they are key to the remainder of this dissertation.

1. Classification and recognition algorithms based on different feature sets and similarity metrics have diverse strengths and weaknesses — they fail and succeed under different conditions.

2. The probabilistic classification estimate from multiple classifiers can be combined, but the properties of the experts dictate the appropriate combination operator. For example, the Bayesian Product Rule assumes the experts are independent, while the Mean Sum Rule assumes the experts represent an independent set of unbiased estimators.

3. Classifiers can be turned into recognition algorithms by searching in pose parameter space using the classifier's similarity metric as an objective function.

4. Pose estimates from multiple recognition algorithms can be combined, but the properties of the pose estimates dictate the appropriate estimate combination algorithm.

5. Previous research and development shows the Condensation Filter is a flexible method for combining pose estimates and the filter makes few assumptions about the underlying process.

Using these observations and the other material reviewed in this chapter, the next chapter describes this dissertation's central research questions, hypotheses, and evaluation criteria.

Chapter 3

Theory

The observations at the end of the previous chapter were never intended to support a unified approach to object recognition and they originated from distinctly different approaches to a variety of problems. However, the observations can be used to define a set of goals for object recognition.

1. Probabilistic information should guide fusion to reduce information loss induced by discrete decision boundaries.

2. Non-parametric representations should be used to avoid overly restrictive and frequently invalid parametric assumptions. For example, the strict parametric assumptions of the Kalman Filter must be avoided, and functions defined in a piece-wise or look-up table form should be accommodated.

3. Recognition cues from low-level features, similarity metrics from classification algorithms, and pose estimates from recognition algorithms should all be fused into a common representation.

In short, an object recognition algorithm should fuse recognition pose estimates, classification similarity metrics, and low-level features to improve the discrimination, precision, and accuracy of object recognition. This chapter describes the theoretical basis for extensions to the Condensation Filter that accomplish these goals. The extensions, with very minor modifications, also provide a general algorithm for supervised learning of the non-parametric representations of uncertainty.

This chapter extends and combines the ideas in the previous chapter to provide three models of the Extended Condensation Filter — first as a statistical filter, then as an Evolution Strategy, and finally as pseudo-code. The pseudo-code model provides step-by-step detail of the algorithm and assists replication of the results presented in Chapter 5. When viewed as a statistical filter, the Extended Condensation Filter is similar to the Condensation Filter, except a more diverse range of inputs are transformed into a common probabilistic representation and the next generation is selected only from the current generation. When viewed as a search algorithm, the Extended Condensation Filter is similar to an Evolution Strategy, except the use of probabilities to weight the particles is intended to make the population resemble the underlying conditional probability for the location of the target object. Both models provide insight about the extended filter, but neither provides the detail needed for replication of the experiments described in Chapter 5. To support replication, the third section provides step-by-step pseudo-code for the training and operation of the filter. Finally, an example at the end of this chapter provides a very simple illustration of an Extended Condensation Filter implementation.

3.1 Statistical Filter Model

Statistical filters estimate the state of a stochastic process based on discrete measurements. Statistical filters can be placed into one of three categories; parametric, non-parametric, or hybrids. Parametric statistical filters, such as the Kalman Filter, require parametric representations of the uncertainty associated with measurements and represent the state of the stochastic process in parametric form [Kal60]. Non-parametric filters, such as the Condensation Filter, do not require parametric representations and generally use discrete populations to represent the process state [Gor97, IB98]. Hybrid filters, such as the mixture of Gaussians Filter, use populations of parametric representations [Pao94]. The goals stated at the beginning of this chapter require independence from any specific parametric representation, so this dissertation will focus on non-parametric statistical filters.

As discussed in the background chapter, the Condensation Filter was one of the first non-parametric statistical filters applied to computer vision [IB96]. To apply the Condensation Filter to object recognition, a population of particles is used, where each particle represents a target object pose $\vec{S}$. In the general non-deformable case, $\vec{S}$ is the six-dimensional pose vector $(x, y, z, \theta, \phi, \rho)$ containing three translation and three rotation parameters. Fewer dimensions can be used for simpler problems and more dimensions can be used to represent

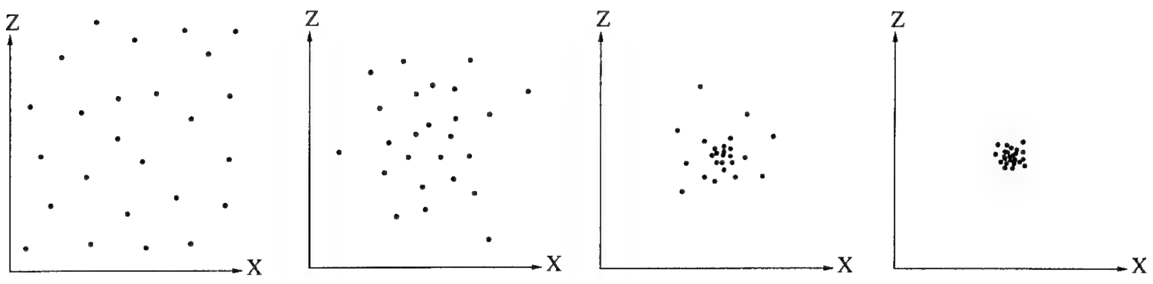


Figure 3.1: The condensation of particles to a dense cluster. A dense cluster is the result of a sequence of precise measurements.

the degrees-of-freedom for a deformable object. In any of these cases, each particle is assigned the conditional probability $P(\vec{S} = \vec{S}^{True}|\vec{s})$ that its pose $\vec{S}$ is near the pose of the true target object $\vec{S}^{True}$, given the measurement $\vec{s}$, where $\vec{s}$ is an estimate of the target object pose provided by some object recognition algorithm.

When a measurement $\vec{s}$ is provided by an object recognition algorithm, each particle's pose $\vec{S}$ is compared to the measurement $\vec{s}$. The uncertainty model associated with the object recognition algorithm that provided the measurement is used to compute $P(\vec{S} = \vec{S}^{True}|\vec{s})$. Generally, the probability $P(\vec{S} = \vec{S}^{True}|\vec{s})$ decreases as the distance between the particle and the measurement increases. The Gaussian function

$$P(\vec{S} = \vec{S}^{True}|\vec{s}) = \frac{e^{\frac{-(\|\vec{S}-\vec{s}\|)^2}{\sigma^2}}}{\sigma\sqrt{2\pi}} \tag{3.1}$$

is a common way to represent this relationship, where $\|\ldots\|$ is the vector magnitude and σ is the standard deviation associated with the measurement source. Object recognition algorithms may also provide a similarity measurement w comparing the pose estimate $\vec{s}$ and the target model. In the Gaussian function, σ may be a function of the similarity metric w, if w is provided. The probability $P(\vec{S} = \vec{S}^{True}|\vec{s}, w)$ is used to weight each particle in the filter. As a result, the weights and positions of the particles are influenced by each measurement and the distribution associated with each measurement.

During the particle reproduction step, this particle weighting causes the percentage of particles with lower probability of being near the target object, lower $P(\vec{S} = \vec{S}^{True}|\vec{s}, w)$ values, to decrease and the percentage of particles with higher probability of being near the target object to increase. After several filter iterations, the particles condense in areas with higher probability of being near the target object. In search terminology, the particles with higher fitness values are reproduced more often than particles with lower fitness values.

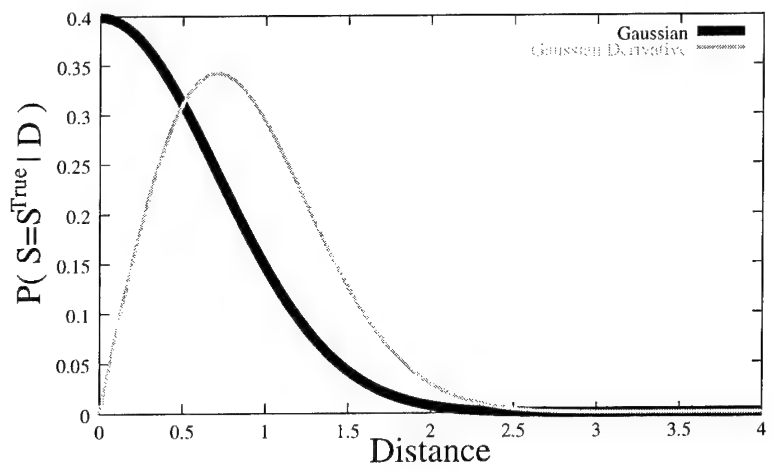


Figure 3.2: Simple Gaussian commonly used to represent $P(\vec{S} = \vec{S}^{True}|(\|\vec{s} - \vec{S}\|)$ and the derivative of the Gaussian

One iteration is one pass through the steps shown in Figure 1.7 on page 10. Figure 3.1 shows an example of particle condensation in the two-dimensional pose space (x, z). After several iterations of the filter, the density of the particles per unit of pose space begins to resemble the pose space probability density function [IB96, IB98]. The size of the cluster will be proportional to the standard deviation of the uncertainty model associated with the measurements. The term "proportional low-pass filter" accurately describes the response of these dense clusters to additional measurements.

3.1.1 Proportional Low-Pass Filter

To show why the Condensation Filter is accurately described as a proportional low-pass filter, consider the following. Assume a Condensation Filter receives a one-dimensional pose measurement $\vec{s}$ per iteration of the filter. Also assume the uncertainty associated with the measurement is modeled by the Gaussian function shown in Figure 3.2, where the independent axis is the distance of the particle's pose from the measurement $\|\vec{s} - \vec{S}\|$, the dependent axis is the probabilistic particle weight $P(\vec{S} = \vec{S}^{True} \mid \|\vec{s} - \vec{S}\|)$, and σ is 1.0.

Reproduction of a particle is not controlled by the absolute weights generated by this Gaussian, but the relative weights of the particles. The probability of selecting a single particle in the current generation to create a single particle in the next generation is the weight of the particle divided by the total weight of all particles

$$P_{repro} = \frac{P(\vec{S} = \vec{S}^{True})}{\sum_{i=0}^{N} P(\vec{S}_i = \vec{S}^{True})} \tag{3.2}$$

where there are N particles in the entire filter population. Since each particle selection is performed with replacement, the probability of selecting a single particle in the current generation to create any of the N particles in the next generation is

$$P_{repro}^{N} = \left(\frac{P(\vec{S} = \vec{S}^{True})}{\sum_{i=0}^{N} P(\vec{S}_i = \vec{S}^{True})} \right)^{N} \tag{3.3}$$

Due to the properties of the Gaussian, the relative weighting within a group of particles near the measurement is very different from the relative weighting within a group of particles far from the measurement.

To illustrate this point, consider the derivative of the Gaussian shown in Figure 3.2. The derivative of the Gaussian is zero at the measurement, increases to a maximum when the distance between the measurement and the particle pose is slightly less than σ, then approaches zero again as the distance increases.

To illustrate the effect of the derivative of the Gaussian on the filter population, consider two particles $\vec{S}_0$ and $\vec{S}_1$ in the Condensation Filter. Assume the two points remain at a constant distance from each other, relative to the measurement

$$||\vec{S}_0 - \vec{s}|| - ||\vec{S}_1 - \vec{s}|| = c \tag{3.4}$$

If the mean distance of these two particles from the measurement, relative to σ, is very small

$$\frac{||\vec{S}_0 - \vec{s}|| + ||\vec{S}_1 - \vec{s}||}{2} << \sigma \tag{3.5}$$

then their weights and their chances of appearing in the next generation of the population will be very similar, since the derivative of the Gaussian near the measurement is very small. If the pair of particles are moved away from the measurement, to the area where the derivative is large, the relative weight of these two particles will be very different — the closer particle will have a much greater chance of appearing in the next generation of the population than the particle only slightly further away. If we move the particles even further away from the measurement, the relative difference in the weights will be small again.

In the one-dimensional pose space, shown in Figure 3.2, there is a band, centered at a value slightly less than σ, where the filter particle population is the most responsive. In some sense, Figure 3.2 is a plot of the velocity of the cluster toward a new measurement as a function of the mean distance between the cluster and the measurement. When a cluster is far away from a series of measurements, the cluster will move slowly toward the measurements, then move faster as the cluster approaches the measurements. Once the cluster is well within σ of the measurements, it responds slowly again. The phrase proportional low-pass filter describes the Condensation Filter's slow response to measurements that are far away, rapid response to measurements that are near σ, and slow response to measurements that are within the uncertainty associated with the source of the measurement. The same is true in higher-dimensional pose spaces where the one-dimensional band becomes a two-dimensional ring, a three-dimensional sphere, and so on.

The results are the same, but only in a piece-wise fashion, if we assume multiple new measurements are received per iteration of the filter. The clusters of particles respond to each new measurement consistent with the proportional low-pass filter model. However, the aggregate response of a cluster to multiple measurements is less predictable due to the interaction of multiple proportional low-pass filters.

My description of the Condensation Filter as a proportional low-pass filter assumes the Gaussian is used to represent uncertainty, but the Condensation Filter can accommodate any parametric uncertainty function. Other functions are not guaranteed to have a derivative that generates the proportional low-pass behavior. To generate the proportional low-pass behavior, duplicating the derivative of the Gaussian is not necessary. The important feature is a derivative that is large near the measurement and grows proportionally smaller further from the measurement. To the degree that a function achieves this shape, it will generate the desirable low-pass proportional behavior. The shape of the derivative of the uncertainty function can be used to predict the response of the filter to individual measurements.

3.1.2 Extensions

The Condensation Filter is a powerful and general statistical filter. However, three critical exceptions prevent the Condensation Filter from meeting the goals stated at the beginning of this chapter.

1. The parametric representations typically used with the Condensation Filter, such as Equation 3.1, are not guaranteed to accommodate an arbitrary collection of object recognition algorithms.

2. Classifiers measure similarity. They do not provide the pose estimates required by a traditional Condensation Filter.

3. Low-level feature extractors provide diverse vectors of features. They do not provide the pose estimates required by a traditional Condensation Filter.

Parametric functions, such as Equation 3.1, are usually used to represent the uncertainty associated with a particle pose, but parametric representations are not required. Non-parametric filters accommodate very general representations, such as functions defined piece-wise or using discrete look-up table. As long as the uncertainty $P(\vec{S} = \vec{S}^{True}|\vec{s})$ can be estimated for any arbitrary object pose estimate $\vec{S}$ and any arbitrary measurement $\vec{s}$, the representation is suitable. To eliminate these exceptions and accommodate the goals at the beginning of this chapter, the following extensions to the Condensation Filter will be described in the next three sections.

1. A general method, based on Bayes' Theorem, will be used to estimate $P(\vec{S} = \vec{S}^{True}|\vec{s}, w)$, for any particle's pose $\vec{S}$ in the filter, given an object recognition algorithm's pose estimate $\vec{s}$ and associated similarity metric w. Neither the Condensation Filter nor Bayesian learning are new, but the combination applied to object recognition has not been reported in the literature.

2. A general method, based on Bayes' Theorem, will be used to estimate $P(\vec{S} = \vec{S}^{True})$, for any particle pose $\vec{S}$ in the filter, given the similarity w of the unknown image and target model. This approach, based on the render-match algorithm [SB01], has not been reported in the literature.

3. A general method, based on Bayes' Theorem, will be used to estimate $P(\vec{S} = \vec{S}^{True})$, for any particle pose $\vec{S}$ in the filter, given a feature vector $\vec{f}$. This approach has not been reported in the literature.

The Extended Condensation Filter is shown in Figure 3.3. The top dotted box, labeled "Recognition", shows the use of object recognition pose estimates $\vec{s}$ to weight particles. The middle dotted box, labeled "Classification", shows the use of a classification algorithm's similarity metric w to weight particles. The bottom box, labeled "Low-Level Feature Extraction", shows how low-level feature vectors $\vec{f}$ are used to weight particles. The next three sections describe how a look-up table in each portion of the diagram is derived and applied.

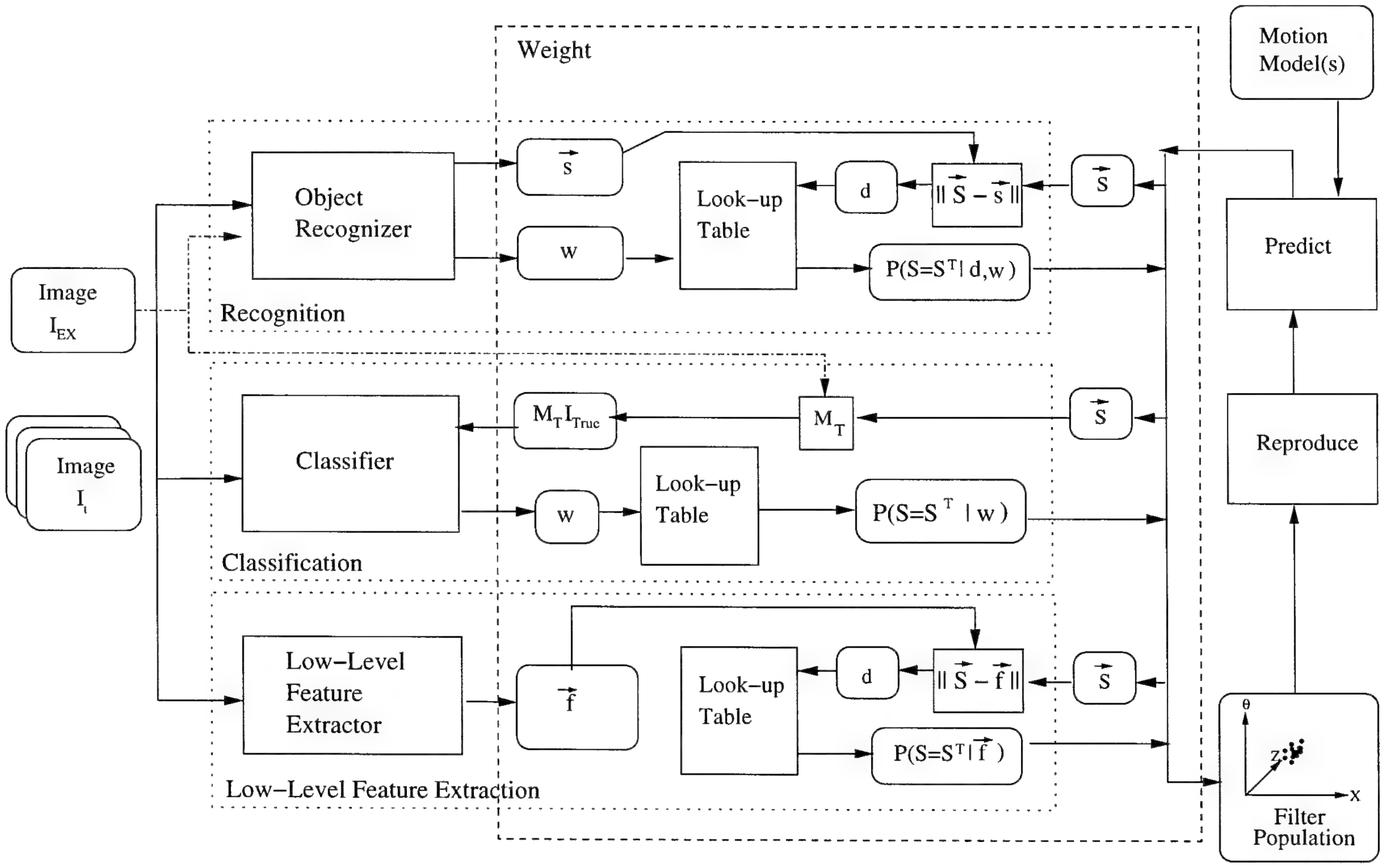


Figure 3.3: The Extended Condensation Filter

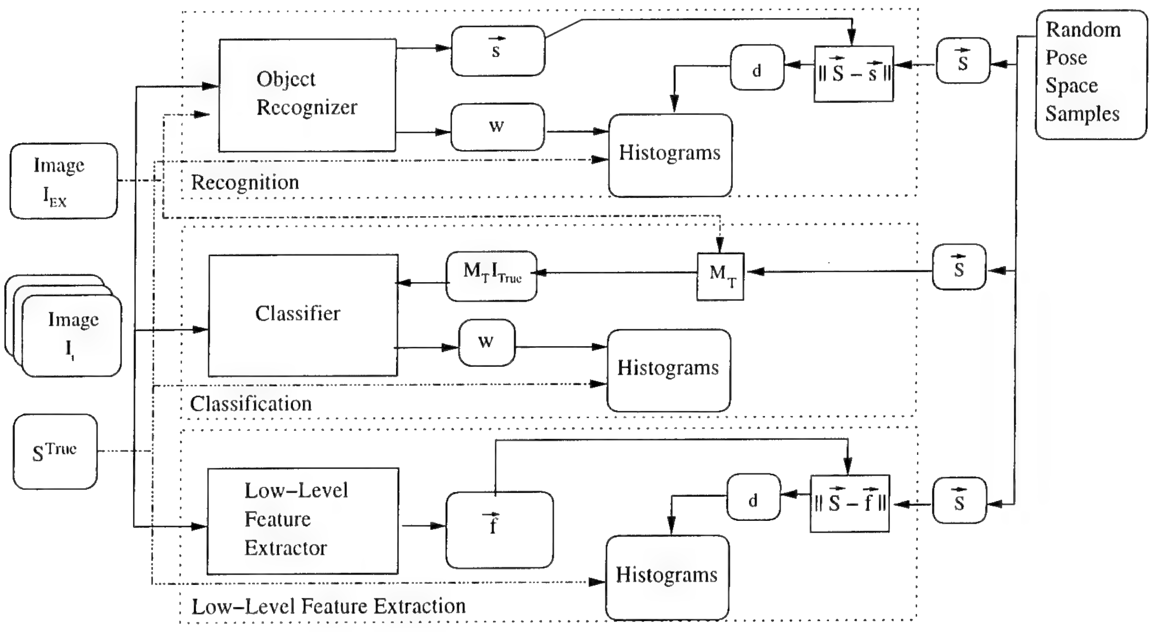


Figure 3.4: Training the Extended Condensation Filter

3.1.3 From Pose Estimate to Probability Estimate

For the Extended Condensation Filter, look-up tables provide flexible representations of conditional probability that replace traditional parametric representations. As shown in the top dotted box of Figure 3.3, the Extended Condensation Filter requires a function

$$P(\vec{S} = \vec{S}^{True} | \vec{s}, w) = F(\vec{S}, \vec{s}, w) \tag{3.6}$$

to transform the pose estimate $\vec{s}$ and similarity metric w provided by a recognition algorithm into probability estimates $P(\vec{S} = \vec{S}^{True} | \vec{s}, w)$.

As established in the goals at the beginning of the chapter, parametric representation are too restrictive. However, a look-up table can provide a flexible discrete representation which meets the goals. A discrete table look-up representation can be built by applying Bayes' Theorem.

$$P(\vec{S} = \vec{S}^{True} | \vec{s}, w) = \frac{P(\vec{s}, w | \vec{S} = \vec{S}^{True}) P(\vec{S} = \vec{S}^{True})}{P(\vec{s}, w)} \tag{3.7}$$

where the values $P(\vec{s}, w | \vec{S} = \vec{S}^{True})$, $P(\vec{S}^{True})$, and $P(\vec{s}, w)$ can be estimated empirically by running the object recognition algorithm on a large number of examples where the ground truth pose $\vec{S}^{True}$ is known. This training phase is shown in the top dotted box of Figure 3.4.

The availability of many examples where the ground truth pose $\vec{S}^{True}$ is known can be an overwhelming requirement. In some applications, large numbers of training examples may not be available. In other cases identifying the ground truth for each example can be tedious or even impossible. For the implementation described in the next chapter, large numbers of examples with known ground truth are easily generated using a synthetic environment. In environments where many examples with known ground truth are not available, synthetic environments may provide a way to simulate the problem environment. Synthetic test and training environments are discussed again in the next chapter.

3.1.3.1 Learning from Pose Estimates

During the training phase it is not necessary to apply Bayes' Theorem to every possible particle pose $\vec{S}$ and measurement $\vec{s}$. Instead, the one dimensional Euclidean distance

$$d_{\vec{s}} = ||\vec{s} - \vec{S}|| \tag{3.8}$$

can be used to condition the probability for the Extended Condensation Filter. If orientation is included, then a second distance measure in rotation space is required or rotation and translation must be normalized and combined in some manner — one method for comparing translation and rotation, called the scaled Euclidean distance, is discussed in Chapter 4. This formulation of the simple distance measure $d_{\vec{s}}$ assumes the camera-relative position of $\vec{s}$ and $\vec{S}$ are irrelevant. This is consistent with the assumption that the probability is isotropic with respect to the world-relative pose. It is possible for this assumption to be incorrect, but recording the data necessary to model the entire space of camera-relative poses is overwhelming given current computer resources. Using the distance $d_{\vec{s}}$ in place of $\vec{s}$ and $\vec{S}$, Bayes' Theorem becomes

$$P(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w) = \frac{P(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})P(\vec{S} = \vec{S}^{True})}{P(d_{\vec{s}}, w)} \tag{3.9}$$

Equation 3.9 is a revision of Equation 3.7 which includes the isotropic assumption.

As shown in Figure 3.4, the probability function can be empirically estimated using a large number of images, where the target object position S^{True} is known. Using these images, the object recognition algorithm is used to estimate the target object pose $\vec{s}$ and associated similarity w.

This learning process requires nearly the same information as the operation of the Extended Condensation Filter itself. Therefore, with only minor modifications, the same software implementation can serve as the learning algorithm and the filter.

Next, as shown at the top right corner of Figure 3.4, a large number of pose vectors $\vec{S}$ are sampled from the entire pose space visible to the camera. These random pose vectors represent the range of poses the Extended Condensation Filter's particles may sample and they should be bounded by the intrinsic limitations of the sensors, such as field of view and resolution, and the parameters for the problem domain.

For the sake of illustration, assume 1000 uniformly random poses $(\vec{S}_0, \ldots, \vec{S}_{999})$ are generated throughout the pose space for each measurement $\vec{s}$. Computing $d_{\vec{s}}$ for all 1000 random poses provides the data required to populate two two-dimensional histograms. One histogram counts all the samples, while the other histogram counts only the samples whose pose is near the target object pose $\vec{S} = \vec{S}^{True}$. Both histograms have the same set of discrete bins — one axis of the bins is $d_{\vec{s}}$ and the other axis is w. This data can be used to estimate the three parameters required to derive $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w)$ using Bayes' theorem.

1. The histogram counting all the random poses can be used to estimate $P(d_{\vec{s}}, w)$ by dividing the number of poses in each bin by the total number of random poses. The term $\hat{P}(d_{\vec{s}}, w)$ indicates an empirical estimate of $P(d_{\vec{s}}, w)$.

2. The histogram counting the random poses that fall near the target object $\vec{S} = \vec{S}^{True}$ can be used to estimate $P(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})$ by dividing the number of poses in each bin by the total number of random poses near the target pose.

3. The total number of random poses in the second $\vec{S} = \vec{S}^{True}$ histogram divided by the total number of random poses can be used to estimate $P(\vec{S} = \vec{S}^{True})$.

These three empirically estimated probabilities provide everything required to estimate $P(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w)$. Using Bayes' Theorem

$$\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w) = \frac{\hat{P}(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})\hat{P}(\vec{S} = \vec{S}^{True})}{\hat{P}(d_{\vec{s}}, w)} \tag{3.10}$$

where the hat over the probability indicates an empirical estimate of the actual probability.

Additional confidence in the estimate is provided by repeating this process using additional random images. Each new image and associated set of 1000 samples $(\vec{S}_0, \ldots, \vec{S}_{999})$

improves the accuracy of $\hat{P}(\vec{S} = \vec{S}^{True})$ and $\hat{P}(d_{\vec{s}}, w)$. However, only the small fraction of points that randomly fall on the target object pose contribute to the accuracy of $\hat{P}(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})$. [3]

To increase the number of samples in the histogram estimating $\hat{P}(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})$, additional samples can be made where $\vec{S} = \vec{S}^{True}$. Since the ground truth target pose S^{True} is known, the target object pose can be purposely sampled to increase the confidence in the estimate. For example, an additional 1000 random samples $\vec{S}$ can be added to the second histogram, where $\vec{S}$ is known to be near $\vec{S}^{True}$. Purposely sampling the pose space near the target object quickly improves the accuracy of $\hat{P}(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})$.

Finally, the estimates of $\hat{P}(d_{\vec{s}}, w)$, $\hat{P}(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})$, and $\hat{P}(\vec{S} = \vec{S}^{True})$ are combined using Bayes' rule to build the look-up table

$$\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w) = Lookup(d_{\vec{s}}, w) \tag{3.11}$$

3.1.3.2 Using Pose Estimates

After the look-up table is available, the Extended Condensation Filter can be run on unknown images without any knowledge of the ground truth — as shown in the top dotted box of Figure 3.3. The measurement pose $\vec{s}$ and the particle pose $\vec{S}$ are used to compute $d_{\vec{S}}$. The values $d_{\vec{S}}$ and w are used to look up $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w)$. Finally, the probability is used to weight the particle during the filter's reproduction step.

This use of the Condensation Filter is typical with the exception that the conditional probability $P(d_{\vec{s}}, w|\vec{S} = \vec{S}^{True})$ is learned and represented using a look-up table, rather than assuming some relationship, such as a Gaussian, and representing it parametrically. This application of Bayes' Theorem allows the use of diverse recognition algorithms without any assumption about the features or metrics. However, the approach requires extensive training examples with known ground truth.

3.1.4 From Classification to Probability Estimate

As shown in the middle dotted box of Figure 3.3, the Extended Condensation Filter requires a function

$$P(\vec{S} = \vec{S}^{True}|w) = F(w) \tag{3.12}$$

[3] $P(\vec{S} = \vec{S}^{True})$ is on the order of 10^{-3} in experiments performed later in this dissertation.

to transform classification similarity metrics w to probability estimates $P(\vec{S} = \vec{S}^{True}|w)$. The approach used for object recognition pose estimates can be adapted to classifiers. To use a classifier to weight a filter particle, the pose $\vec{S}$ must be used to compare the example target image I_{EX} to the unknown image I_t acquired at time t. Intuitively, the weight of the filter particle is proportional to the similarity between the target image and the unknown image, if the target image were located at the pose $\vec{S}$.

Every pose $\vec{S}$ defines a unique affine transformation matrix $M_{\vec{S}}$. The transformation matrix can be used to project the target image into pose space, as discussed in Section 2.4. The transformed image $M_{\vec{S}}I_{EX}$ provides an estimate of the target appearance, if it were located at the camera relative pose $\vec{S}$. Given the transformed target image $M_{\vec{S}}I_{EX}$, any classifier can be used to measure the similarity w between the transformed target $M_{\vec{S}}I_{EX}$ and the relevant portion of the unknown image I_t.

Three-dimensional target models can be used in place of two-dimensional target images, and transformed using the particle's pose $\vec{S}$. For many problems, three-dimensional models provide a superior estimate of the target object appearance [SB01]. Only for objects that are well modeled by a two-dimensional image, such as a flat plate or one side of a cube, is the transformed two-dimensional target image a fairly accurate estimate. For this dissertation, only two-dimensional models are tested. However, this is not a limitation of the Extended Condensation Filter, just a limitation of my implementation. The disadvantages of using simple two-dimensional target models to represent three dimensional objects is investigated later in this dissertation.

Since the two or three-dimensional target model can be projected and compared to the unknown image, a similarity metric w is available. Now classifier similarity measures, just like recognition algorithm pose estimates, can be used to infer conditional probability. The conditional probability can be represented by a look-up table based on an empirical estimate of $P(\vec{S} = \vec{S}^{True}|w)$. Using Bayes' Theorem

$$P(\vec{S} = \vec{S}^{True}|w) = \frac{P(w|\vec{S} = \vec{S}^{True})P(\vec{S} = \vec{S}^{True})}{P(w)} \tag{3.13}$$

where the values $P(w|\vec{S} = \vec{S}^{True})$, $P(\vec{S}^{True})$, and $P(w)$ can be estimated empirically by running the classifier on a large number of representative examples, where $\vec{S}^{True}$ is known. This training phase is shown in the middle dotted box of Figure 3.4 and explained further in the next section.

3.1.4.1 Learning from a Classifier

Training using a classifier starts just like training using an object recognition algorithm — the visible pose space is sampled. Each of these pose vectors defines an affine transformation matrix $M_{\vec{S}}$ that defines an affine transformation of the target image $M_{\vec{S}} I_{EX}$.

For the purposes of illustration, assume 1000 random poses $(\vec{S}_0, \ldots, \vec{S}_{999})$ are generated uniformly throughout the pose space, just like the object recognition example. Computing w for all 1000 random poses provides data that can be used to populate two one-dimensional histograms. Just like the use of object recognition pose estimates, one histogram counts all the samples, while the other histogram counts only the samples whose pose is near the target object pose $\vec{S} = \vec{S}^{True}$. Both histograms have the same set of bins that represent discrete values of w. This data can be used to estimate the three parameters required to derive $\hat{P}(\vec{S} = \vec{S}^{True}|w)$ using Bayes' Theorem.

1. The histogram counting all the random poses can be used to estimate each $P(w)$ by dividing the number of poses in each bin by the total number of random poses. The term $\hat{P}(w)$ indicates an empirical estimate of $P(w)$.

2. The histogram counting the random poses that fall on the target object $\vec{S} = \vec{S}^{True}$ can be used to estimate $P(w|\vec{S} = \vec{S}^{True})$ by dividing the number of poses in each bin by the total number of random poses near the target pose.

3. The total number of random poses in the second $\vec{S} = \vec{S}^{True}$ histogram divided by the total number of random poses can be used to estimate $P(\vec{S} = \vec{S}^{True})$.

Once the histograms are available, Bayes' Theorem can be used to estimate the probability an arbitrary pose $\vec{S}$ is near the ground truth $\vec{S}^{True}$, given the similarity w

$$\hat{P}(\vec{S} = \vec{S}^{True}|w) = \frac{\hat{P}(w|\vec{S} = \vec{S}^{True})\hat{P}(\vec{S} = \vec{S}^{True})}{\hat{P}(w)} \tag{3.14}$$

Additional confidence can be obtained by using many images and purposely sampling the poses near the target object, just like the algorithm described in the last section to learn the conditional probability for the object recognition algorithms.

Finally, the estimates of $\hat{P}(w)$, $\hat{P}(w|\vec{S} = \vec{S}^{True})$, and $\hat{P}(\vec{S} = \vec{S}^{True})$ are combined using Bayes' rule to build the look-up table

$$\hat{P}(\vec{S} = \vec{S}^{True}|w) = Lookup(w) \tag{3.15}$$

3.1.4.2 Using a Classifier

After the look-up table is built, the Extended Condensation Filter can be run on unknown images without any knowledge of the ground truth – as shown in Figure 3.3. Given a particle from the filter population, the particle's pose $\vec{S}$ is used to create an affine transformation matrix $M_{\vec{S}}$. The affine transformation matrix is used to transform the example target image I_{EX}. The transformed example target image $M_{\vec{S}} I_{EX}$ is compared to the unknown image to create the similarity metric w. The similarity metric w is used to look up the estimated probability $P(\vec{S} = \vec{S}^{True}|w)$. Finally, the probability is used to weight the particle during the filter's reproduction step.

This use of the Condensation Filter represents a departure from previous work. Rather than evaluate a particle as a function of its distance from some estimate $\vec{s}$, each particle's pose is evaluated directly using the transformation matrix and classifier. This approach is more typical of the way classifiers are used to localize pose estimates using a search algorithm [HR92, RB95, SB01]. This approach also gives the Extended Condensation Filter active control of where classifier similarity measurements are made — classifier measurements are only made at each particle pose and the filter controls the particle pose.

3.1.5 From Low-Level Feature to Probability Estimate

As shown in the bottom dotted box of Figure 3.3, the Extended Condensation Filter requires a look-up table to transform low-level features to probability estimates. Running a low-level feature extractor on an image may generate a large set of low-level feature vectors. Low-level features can be divided into two types: two-dimensional features on the image plane, such as edgels, and three-dimensional features in pose space, such as range features. This section will address three-dimensional pose space features, then extend the model to include two-dimensional image plane features.

Sets of low-level feature vectors $\{\vec{f}_0, \ldots, \vec{f}_J\}$, where J is the total number of feature vectors extracted from one image, may provide useful information about the probability that an arbitrary pose is the target object $P(\vec{S} = \vec{S}^{True}|\{\vec{f}_0, \ldots, \vec{f}_J\})$. For example, given a set of stereo disparity range features $\{\vec{f}_0, \ldots, \vec{f}_J\}$, what is the probability that a pose $\vec{S}$ is the target object? Intuitively, the target object is probably not between the camera and a range feature. Also, the target is probably not visible behind a range feature. This information is useful because it narrows the space of probable poses for the target object, but the stereo disparity features do not indicate a single most probable location for the target object.

The utility of a low-level feature will depend on the feature set and the problem domain. For example, stereo disparity features may provide little, if any, discrimination in a problem domain which consists of recognizing thin flat targets mounted on a wall. The choice of low-level features is an important design decision. Due to the learning process, a poor low-level feature choice should not degrade the performance of the filter. However, resources could be wasted extracting features which provide little information about the conditional target pose probability.

Figure 3.3 shows the use of a set of feature vectors in the Condensation Filter. Given the pose $\vec{S}$ associated with a particle in the Condensation Filter and a set of feature vectors $\{\vec{f_0}, \ldots, \vec{f_J}\}$, represented in camera-relative coordinates (x, y, z), the particle weight should be a function of the support indicated by the feature vector. Mathematically, the function is

$$P(\vec{S} = \vec{S}^{True} | \{\vec{f_0}, \ldots, \vec{f_J}\}) = F(\vec{S}, \{\vec{f_0}, \ldots, \vec{f_J}\}) \tag{3.16}$$

Just like the use of object recognition and classification information, a training phase is used to build a look-up table representation of this function.

3.1.5.1 Learning from Low-Level Features

Applying Bayes' Theorem, the function for transforming a set of feature vectors to a probability estimate becomes

$$P(\vec{S} = \vec{S}^{True} | \{\vec{f_0}, \ldots, \vec{f_J}\}) = \frac{P(\{\vec{f_0}, \ldots, \vec{f_J}\} | \vec{S} = \vec{S}^{True}) P(\vec{S}^{True})}{P(\{\vec{f_0}, \ldots, \vec{f_J}\})} \tag{3.17}$$

However, estimating this relationship for every possible combination of hundreds or thousands of features is impractical and unnecessary. Instead, distance metrics can use the nearest feature, K nearest features, or some other statistic about the features, such as the centroid. Many such formulations are available in the literature and the choice of a specific distance metric is implementation specific. The choice of a distance measure should be based on both the properties of the low-level feature and the problem domain.

For the purposes of illustration, my model uses the simple Euclidean distance to the nearest feature, as shown in the bottom dotted box of Figure 3.4

$$d_{\vec{f}} = \min_{j=0}^{J}(\|\vec{f_j} - \vec{S}\|) \tag{3.18}$$

Using the minimum distance to condition the probability, Bayes' Theorem becomes

$$P(\vec{S} = \vec{S}^{True}|d_{\vec{f}}) = \frac{P(d_{\vec{f}}|\vec{S} = \vec{S}^{True})P(\vec{S}^{True})}{P(d_{\vec{f}})} \tag{3.19}$$

Just like the use of recognition and classification algorithms, the probability can be estimated using a large number of images where the target object position S^{True} is known. A set of low-level feature vectors $\{\vec{f}_0, \ldots, \vec{f}_J\}$ are extracted for each image. Next, a large number of pose vectors $\vec{S}$ are generated throughout the entire pose space visible to the camera.

For the purposes of illustration, assume 1000 random poses $(\vec{S}_0, \ldots, \vec{S}_{999})$ are generated uniformly throughout the pose space for each set of features $\{\vec{f}_0, \ldots, \vec{f}_J\}$ provided by the low-level feature extraction algorithm. Comparing all 1000 random poses to all the feature vectors to compute $d_{\vec{f}}$ for each random pose, provides data that can be used to populate two two-dimensional histograms. One histogram counts all the samples, while the other histogram counts only the samples whose pose is near the target object pose $\vec{S} = \vec{S}^{True}$. Both histograms have the same set of bins representing the discrete values of $d_{\vec{f}}$. This data can be used to estimate the three parameters required to derive $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{f}})$ using Bayes' Theorem.

1. The histogram counting all the random poses can be used to estimate $P(d_{\vec{f}})$ by dividing the number of poses in each bin by the total number of random poses. The term $\hat{P}(d_{\vec{f}})$ indicates an empirical estimate of $P(d_{\vec{f}})$.

2. The histogram counting the random poses that fall on the target object $\vec{S} = \vec{S}^{True}$ can be used to estimate $P(d_{\vec{f}}|\vec{S} = \vec{S}^{True})$ by dividing the number of poses in each bin by the total number of random poses near the target pose.

3. The total number of random poses in the second $\vec{S} = \vec{S}^{True}$ histogram divided by the total number of random poses can be used to estimate $P(\vec{S} = \vec{S}^{True})$.

Once the histograms are available, Bayes' Theorem can be used to estimate the probability an arbitrary pose $\vec{S}$ is near the ground truth $\vec{S}^{True}$, given the distance to the nearest feature $d_{\vec{f}}$. Using Bayes' Theorem

$$\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{f}}) = \frac{\hat{P}(d_{\vec{f}}|\vec{S} = \vec{S}^{True})\hat{P}(\vec{S} = \vec{S}^{True})}{\hat{P}(d_{\vec{f}})} \tag{3.20}$$

Additional confidence in the estimate can be gained by repeating this process using additional images. However, only the small fraction of points that randomly fall on the target object pose contribute to the accuracy of $\hat{P}(d_{\vec{f}}|\vec{S} = \vec{S}^{True})$. Just like the object recognition and classification problems, the target object pose can be purposely sampled to increase the confidence in the estimate.

Next, the values of $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{f}})$ are used to build the look-up table

$$\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{f}}) = Lookup(\min_{j=0}^{J}(\|\vec{f}_j - \vec{S}\|)) \tag{3.21}$$

3.1.5.2 Using Low-Level Features

After the look-up table is built, the Extended Condensation Filter can be run on unknown images without any knowledge of the ground truth — as shown in the bottom dotted box of Figure 3.3. The set of feature vectors $\{\vec{f}_0, \ldots, \vec{f}_J\}$ and each particle's pose $\vec{S}$ are used to compute the minimum distance $d_{\vec{f}}$. The closest feature vector to the particle pose is used to compute the distance. The value of $d_{\vec{f}}$ is used to look-up the probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{f}})$. Finally, the probability is used to weight the particle during the filter's reproduction step.

This use of the Condensation Filter represents a departure from previous work. Sets of low-level features do not provide direct estimates of the target pose. However, low-level features can be used to infer something about the target object pose. For example, stereo disparity range features only indicate undetermined objects occupy some range of poses. However, poses which appear to contain no object are unlikely to contain the target object. The degree to which they can be used to infer information about the target object depends entirely on the problem domain and the properties of the feature vector. Results of testing an implementation of the Extended Condensation Filter later in this dissertation show low-level features can provide a surprisingly large amount of information about the target object pose.

3.1.5.3 Two and Three-Dimensional Low-Level Features

Throughout this discussion of low-level features, pose vectors $\vec{S}$ and feature vectors $\vec{f}$ are treated as if they always exist in the same camera-relative coordinate system. In some cases, such as the three-dimensional range features provided by stereo disparity, they do.

However, some features are only two-dimensional and they are most-easily represented on the camera's image plane. Such features include the edgels provided by edge detectors and the color matched pixels provided by color histogram back-projection. The location of these features is best represented on the two-dimensional (u, v) image plane because the depth of the feature is unknown.

To place the two-dimensional image plane feature vector and the possibly six-dimensional particle pose vector in the same coordinate system, each particle's pose is projected onto the image plane. This projection computes the (u_S, v_S) point on the image plane where the particle would appear, if the particle were an object with the pose $\vec{S}$. This is the same process frequently used in computer graphics to render images from geometric object models. The (u_S, v_S) location of the particle pose on the image plane and the (u_f, v_f) location of the feature on the image plane are now in the same coordinate system. The distance

$$D_{\vec{f}} = \|(u_S, v_S) - (u_f, u_f)\| \tag{3.22}$$

between the two points can be used, just like the three-dimensional features, to estimate and look-up $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{f}})$.

3.1.6 Combining Probability Estimates

Using the methods in the previous three sections, pose estimates, classification metrics, and low-level features can be separately transformed into a common representation, but the common representation requires a single reference pose $\vec{S}$. This is not a limitation when using the Condensation Filter because the filter relies completely on weights applied to known poses represented by its population of particles. However, the Condensation Filter does require a single weight for each particle. If multiple recognition, classification, and feature extraction algorithms are used, the separate probability estimates must be combined.

The right side of Figure 3.3 shows an iterative series of steps performed on the filter population. Within the weight step, each particle assigned pose $\vec{S}$ is weighted using information provided by the recognition (top dotted box), classification (middle dotted box), and low-level feature extraction (bottom box) algorithms. Each piece of information must be combined to create a single probability or likelihood estimate used as a weight during particle reproduction.

Using the transformations developed in the previous sections, all the information sources can be represented as estimates of the conditional probability $\hat{P}(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N)$, where Δ is a generic symbol representing either the distance between an object recognition pose estimate $\vec{s}$ and the particle pose $\vec{S}$, a classifier similarity metric w, or the distance between the nearest low-level feature $\vec{f}$ and the particle pose $\vec{S}$. Using probabilistic representations allows the filter to capitalize on previous classifier combination results, such as the Bayesian product rule and the mean sum rule discussed in Section 2.3.1 [GZ86, KHDM98, JDM00].

Using the product rule from Equation 2.8 on page 33, the combined probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N)$, becomes

$$\hat{P}(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N) = \frac{\hat{P}(\vec{S} = \vec{S}^{True}) \prod_{i=1}^{N} \hat{P}(\Delta_i|\vec{S} = \vec{S}^{True})}{\prod_{i=1}^{N} \hat{P}(\Delta_i)} \tag{3.23}$$

where $\Delta_0, \ldots, \Delta_N$ represents a combination of N pose estimates, similarity measures, and low-level features. Using this same notation, the mean sum rule from Equation 2.15 on page 36 becomes

$$L(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N) = \frac{\sum_{i=0}^{N} P(\vec{S} = \vec{S}^{True}|\Delta_i)}{N} \tag{3.24}$$

These two rules, when combined with the transformations described in the previous sections, provide two ways to combine diverse information to create a single probability or likelihood estimate (or weight) for each Extended Condensation Filter particle based on the properties of the pose that particle represents. However, each rule represents a set of assumptions about the sources of information. For example, the Bayesian product rule assumes the sources are independent, while the mean sum rule assumes the sources are unbiased estimates. The validity of these assumptions is problem dependent and tested in Chapter 5.

3.1.7 Combining Dynamic Pose Estimates

Up to now, a single unknown image I_t has been assumed. The more general object recognition problem includes environmental motion observed in a time sequence of images —

a movie. The image I_t becomes just one image in this image sequence. Fortunately, the Condensation Filter is designed to accommodate image sequences with moving objects [IB96, IB98]. The single image I_t in Figure 3.3 is replaced by a sequence of images $(I_0, \ldots, I_T)$, where each image I_t is acquired at time t, and motion models are used in a prediction step.

As shown in Figure 3.3, motion models are used to predict the effect of moving objects on the particle population. A simple motion model for non-deformable objects is

$$\begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \rho \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ \phi_0 \\ \theta_0 \\ \rho_0 \end{bmatrix} + \begin{bmatrix} v_x t \\ v_y t \\ v_z t \\ v_\phi t \\ v_\theta t \\ v_\rho t \end{bmatrix} \tag{3.25}$$

where x_0 is the objects position along the x-axis at time $t = 0$ and v_x is the x component of the object's velocity. More general motion models may include high-order terms and stochastic variables. For example, the velocity terms v_x, v_y, v_z, v_ϕ, v_{theta} and v_{rho} can be defined as Gaussian random variables.

To illustrate the use of a motion model in the Condensation Filter, consider the moving camera and stationary target object shown in Figure 3.5. Assume the camera starts at time $t = 0$ at the point (0, 0, -10) and moves along the z-axis with a velocity defined by a Gaussian random variable. The motion model becomes

$$\begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \rho \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -10 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ v_z(1,1)t \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.26}$$

where, v_z is a Gaussian random variable with mean $\mu_{v_x} = 1$ and standard deviation $\sigma_{v_x} = 1$. The camera acquires one new image I_t per second. If the world-relative ground truth object pose is (0, 0, 20, 0, 0, 0) at time t=0, the camera-relative ground truth pose $\vec{S}^{True}$ is (0, 0, 30, 0, 0, 0).

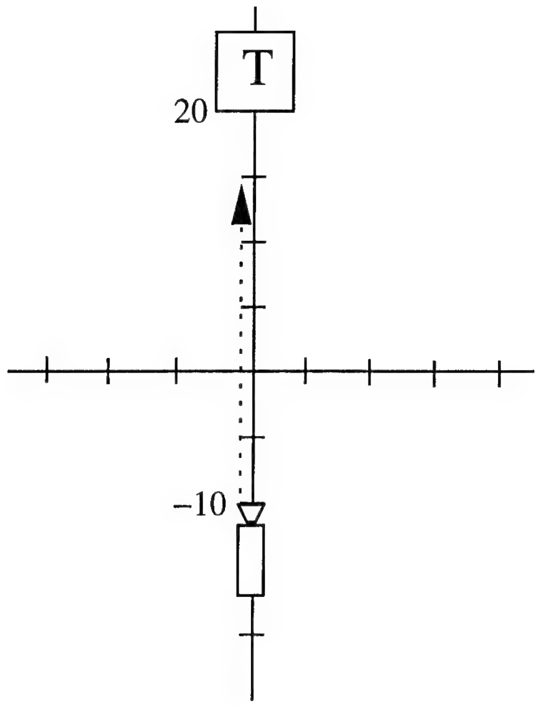


Figure 3.5: Motion Model Example

Since the camera acquires one new image per unit of time and the camera moves forward with a mean velocity of one x-unit per unit of time, the expected value of $\vec{S}^{True}$ in the next frame is (0, 0, 29, 0, 0, 0). The goal is to incorporate this information into the cluster of particles representing the Condensation Filter's current estimate. Ideally, the particles in the Condensation Filter are in a cluster near the point where $P(\vec{S} = \vec{S}^{True})$ is maximum. Based on the motion model in the example, the maximum point is expected to move one unit closer to the camera. Therefore, each filter population particle is moved by adding a Gaussian random number generated using the motion model distribution transformed to camera-relative coordinates

$$\vec{S}_{t+1} = \vec{S}_t + \begin{bmatrix} 0 \\ 0 \\ -v_z(1,1)t \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.27}$$

Figure 1.7 on page 10 shows the predict step where the particles are moved using this probabilistic camera-relative motion model. If two motion models are used, one for the target object and one for the camera, both stochastic vectors must be added to each particle.

The motion model in my example is extremely simple. To illustrate the simplicity of my one-dimensional model, consider a basic 18 dimension Kalman Filter used by aircraft navigation systems. A typical implementation includes six degrees of freedom, three translation and three rotation, and estimates of the position, velocity, and acceleration in each dimension. Such a model usually includes the relationship between all of the components of the model. In other words, the Gaussian estimating the next state for each of the 18 dimensions is a function of all 18 dimensions [WG73]. Nothing indicates either the Condensation Filter or Extended Condensation Filter are incapable of accommodating more complex models. However, to keep my initial implementation simple, I use the one-dimensional motion model shown above.

Using the statistical filter model described to this point, recognition, classification, and low-level feature extraction algorithms can be used to estimate the combined probability, or weight, for each of the particles in the Condensation Filter. The particles are re-weighted after each new image is acquired, allowing the filter to incorporate the predictive information provided by target object and camera motion models. One final step is required to make the Extended Condensation Filter useful — extracting the object pose estimate based on the aggregate state of the filter particles.

3.1.8 Extracting an Estimate

As shown at the bottom right of Figure 3.3, the Condensation Filter's population of particles is intended to provide a discrete representation of the conditional probability for the object pose space. To extract a single most-probable estimate of the current target location requires an analysis of the filter population.

During an early experiment preparing for this dissertation, I identified a disadvantage of Condensation Filters that is seldom addressed in the literature. Despite praising the Condensation Filter's ability to represent multi-modal non-Gaussian distributions, most applications assume the filter particles represent a single Gaussian distribution. Using the single Gaussian assumption, the filter's pose estimate is well represented by the filter population's mean particle pose and standard deviation.

In the Extended Condensation Filter implemented for these early experiments, the uncertainty associated with each of the object recognition algorithms was represented by a Gaussian. As a result, conflicting measurements led to a distribution composed of a mixture of Gaussians. There were time intervals when the filter population was clearly multi-modal. Taking the mean value of the filter population in this state provides a mis-leading pose estimate. The multi-modal population creates a classification problem — which distribution does the random sample represent? If the modes could be separated, then the mean for each mode would be a representative statistic.

In the preliminary study, the problem was solved using a bottom-up hierarchical clustering algorithm [Mur85, DH73]. Unfortunately, hierarchical clustering with hundreds or thousands of particles is computationally demanding. Other clustering algorithms might offer more efficient methods, such as single-pass non-hierarchical clustering or Expectation Maximization trials assuming different numbers of Gaussian distributions, that could be used to assign particles to distributions.

3.2 Evolution Strategy Model

As shown in Table 2.3 on page 49, the Condensation Filter is similar to an Evolution Strategy. Both the Condensation Filter and an Evolution Strategy use weighted random reproduction of a population of potential solutions to focus search on portions of the state space that produce consistently high fitness values. The Condensation Filter and Evolution Strategy also purposely create potentially less fit population members to explore novel portions of the search space.

An Evolution Strategy and Condensation Filter also have a significant difference. The probabilistic functions used by a Condensation Filter cause the aggregate population to model the underlying conditional probability [Gor97]. As a result, statistics about the aggregate population are used to estimate the most probable pose. Typical applications of an Evolution Strategy place less emphasis on modeling the underlying process in favor of an evolutionary mechanism that most rapidly converges to highly fit solutions [BHS91, Whi94, Won98]. As a result, the most fit population members are used to estimate the optimal states. The reproduction and mutation mechanisms of an Evolution Strategy can be made to resemble the Condensation Filter. In this case, an Evolution Strategy would behave much like a Condensation Filter and the population of particles should, according to Gordon and Blake [Gor97, IB98], represent the conditional probability.

The similarities of the Evolution Strategy and Condensation Filter provide an interesting way to examine the effects of combining recognition, classification, and low-level feature extraction algorithms.

3.2.1 Searching in Pose Space

The Extended Condensation Filter moves particles toward the poses $\vec{S}$ that consistently returns the maximum probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N)$ — a search problem.

In the search literature, it is well established that the shape of the fitness function dictates the difficulty of the search problem [MW95]. The Condensation Filter performs search, but the literature rarely discusses the effect the shape of the conditional probability function has on the performance of the filter. Instead, the Condensation Filter literature assumes a very large population of particles — this lightly dismisses the search problem. In practice, Condensation Filters may have relatively small populations of particles, compared to the size of a real-valued pose space, and the conditional probability function may contain large numbers of local minima. Therefore, the shape of the conditional probability function dictates the difficulty of finding the optimal ground truth pose $\vec{S}^{True}$. The example in this section shows how a combination of recognition, classification, and low-level feature information can combine to make the pose space easier to search. The issue of effective search may also be associated with current research on the effective number of particles represented by a given Condensation Filter implementation [Mac00]. However, work in this area is on-going.

To illustrate this point, consider the object recognition problem shown in Figures 3.6. The problem is locating the target object, shown on the left of Figure 3.6, in the unknown image, shown on the right of Figure 3.6. To simplify the example, only the horizontal translation parameter x is unknown. In other words, for this example $\vec{S} = x$.

Three algorithms are run using the images shown in Figure 3.6 to provide information for the search; a Hausdorff-Huttenlocher recognition algorithm with a canonical Gaussian uncertainty model, a pixel correlation classifier with a look-up table representation of uncertainty, and a color-based low-level feature extraction algorithm with a Gaussian repre-

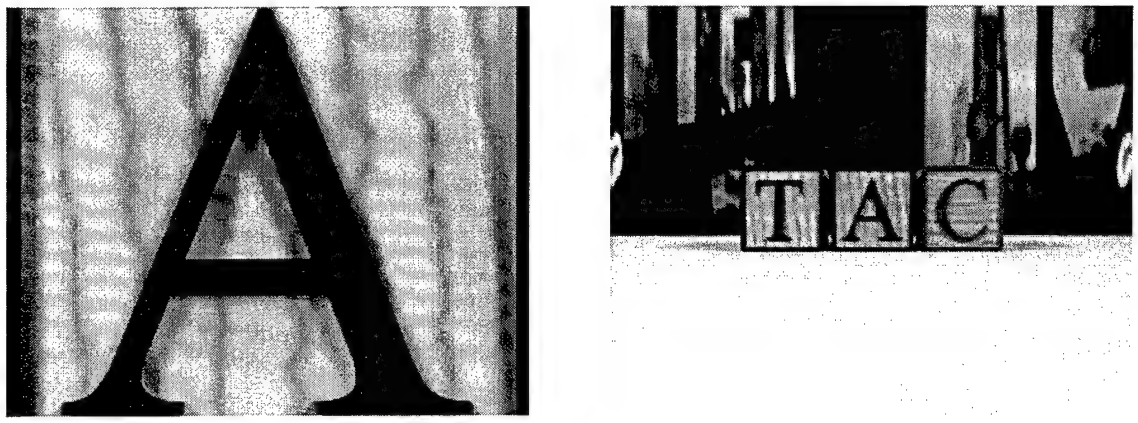


Figure 3.6: Object recognition problem used to illustrate search model

sentation of uncertainty based on the closest feature. These algorithms are described in greater detail later in this dissertation, but the specifics are not required for this example.

Figure 3.7 shows a typical measurement and representation of uncertainty for an object recognition algorithm as a function of x. Figure 3.7 indicates the recognition algorithm has identified the most probable target pose as x = -0.25. From a search point of view, finding the maximum value of the function shown in Figure 3.7 is trivial. However, the Gaussian does not represent the real similarity between the target image and the unknown image as a function of x.

The object recognition algorithm's use of a Gaussian to represent uncertainty artificially over-simplifies the search problem. A classification algorithm provides a more realistic view of the search problem. If the target object is translated to each position x and compared, as shown in 3.8, the result is not as smooth as shown in Figure 3.7. The similarity of the target and unknown images as a function of the pose space variable x contains many local maxima. As a result, the global maximum is harder to find using search.

Low-level feature extraction algorithms are not looking for the target. They only provide a vector of features that may be coincident with the target object, such as pixels with the right color or range features. Low-level feature extraction algorithms only indirectly discriminate between target and non-target objects. As shown in Figure 3.9, using the distance to the nearest feature, as described earlier, may create large plateaus with equal probability, so far as the feature extraction algorithm is concerned. From a search perspec-

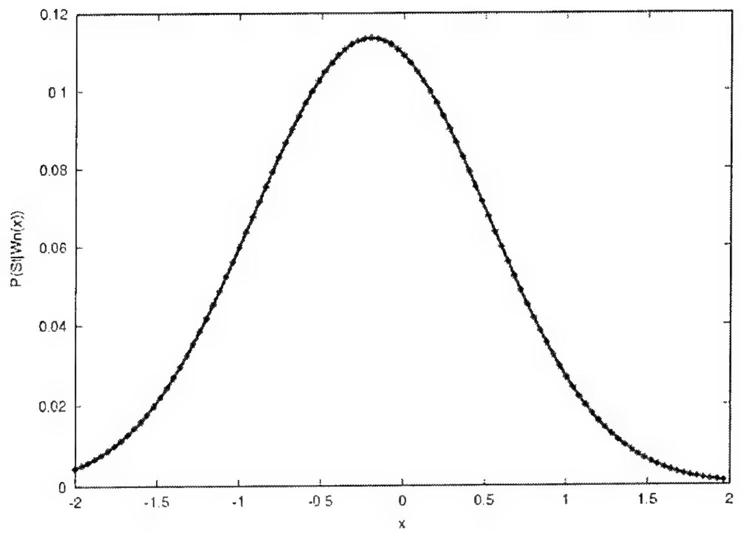


Figure 3.7: Object recognition (Hausdorff-Huttenlocher) uncertainty model, where the uncertainty $\hat{P}(\vec{S} = \vec{S}^{True}|d_{\vec{s}}, w)$ is modeled by a Gaussian. For this example $\vec{s}$ is simply the horizontal translation x.

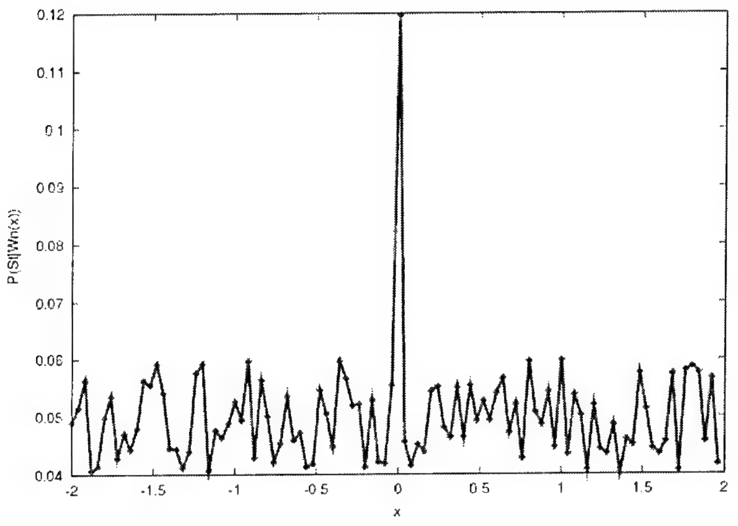


Figure 3.8: Classifier (render-match) uncertainty model, where the uncertainty $\hat{P}(\vec{S} = \vec{S}^{True}|Wn)$, as a function of x, is derived directly from the similarity metric computed at each location x.

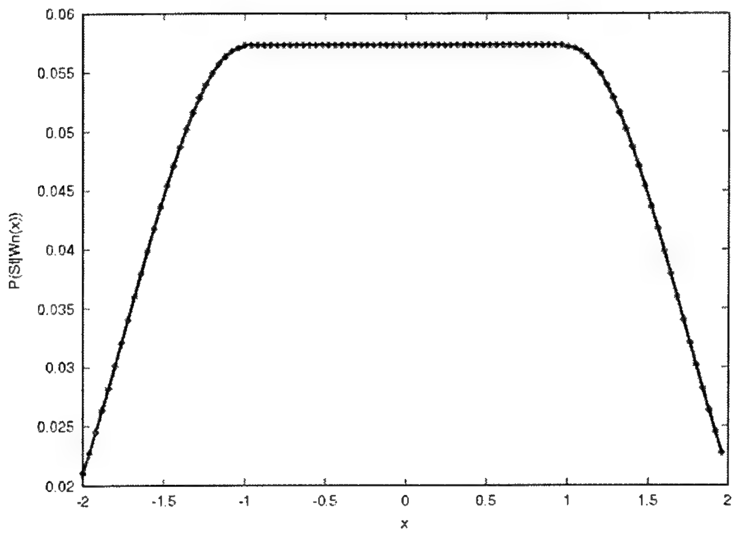


Figure 3.9: Low-level (back-projection color histogram) feature uncertainty model, where the uncertainty $\hat{P}(\vec{S} = \vec{S}^{True}|Wn)$, as a function of x, is modeled by a Gaussian using the closest feature.

tive, low-level features narrow the search, but fail to indicate a pose with the maximum probability of being the target object.

Combining the data shown in Figures 3.7, 3.8, and 3.9 using the Bayesian product rule described earlier creates the plot shown in Figure 3.10. The contribution of the recognition, classification, and low-level feature data is clearly visible in the combined plot. The recognition and low-level feature data have created an overall gradient in the probability estimate as a function of x, while the classifier has contributed the real similarity measure. Due to the gradients added by the recognition and low-level feature extraction algorithms, the combined probability function is easier to search than the conditional probability using only the classification algorithm.

Note the difference in the target pose indicated by the classifier and the recognition algorithm. The recognition algorithm pose estimate is approximately x = -0.25, while the classifier indicates the greatest similarity is approximately $x = 0$. The difference could be an error from the recognition algorithm, the classifier, or both. If the three algorithms continue to provide these estimate, the disagreement between the recognition estimate and the classifier similarity scores is likely to be resolved within a few generations because particles in the vicinity of $x = 0$ will have a very high probability of reproduction. As a result, the pose space near $x = 0$ will be sampled far more heavily. If the difference

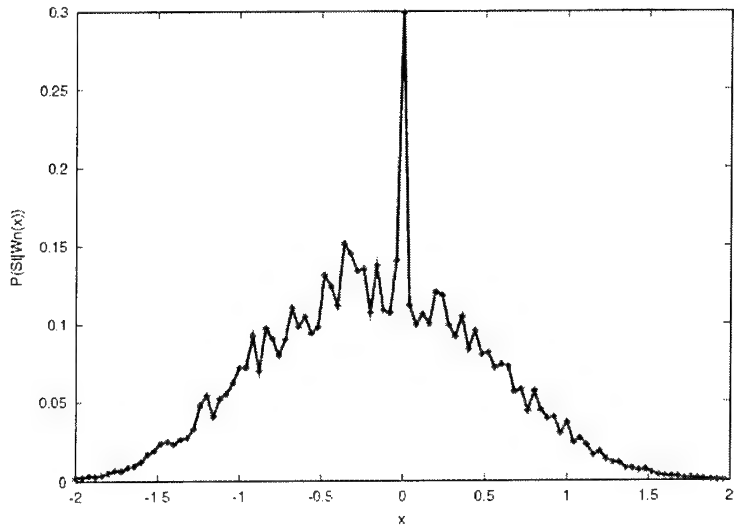


Figure 3.10: Combined product rule estimate of $\hat{P}(\vec{S} = \vec{S}^{True}|Wn)$ using the data from Figures 3.7, 3.8, and 3.9.

persists, the particles will be more dispersed. The dispersion of particles is controlled by the uncertainty resulting from the difference in the pose estimates.

3.3 Extended Condensation Filter Algorithm

The previous sections provide descriptive models of the Extended Condensation Filter. The next chapter tests a specific implementation of the algorithm in two synthetic problem domains. The following step-by-step algorithmic description bridges the gap between the descriptive models and the implementation. The step-by-step algorithmic description also provides the information needed to replicate the experiments in this dissertation.

The first algorithm in this section describes the filter's learning phase, shown in Figure 3.4 on page 61, where the data required to build the look-up tables is collected. The second section provides a step-by-step algorithm for the filter's object recognition phase, shown in Figure 3.3 on page 60, where the learned look-up tables are used to perform object recognition using the product rule. For brevity, a step-by-step model using the mean sum rule has not been provided. However, the mean sum rule can be directly substituted where the Bayesian product rule is used in the following examples.

3.3.1 Building the Probability Estimate Look-Up Table

1. Acquire target model, including the example target image I_{EX}, the target parameters (range to target), and the sensor parameters (field of view, resolution).

2. Repeat the following steps to initialize learning for each object recognition algorithm.

   2.1. Create a histogram for estimating the distribution of all particle poses. Each bin is assigned a range for values near the distance $d_{\vec{S}}$ and similarity w. Set each bin's count to zero.

   2.2. Create a histogram for estimating the distribution of particles whose pose is near the ground truth $\vec{S} = \vec{S}^{True}$. Each bin is assigned a range for values near the distance $d_{\vec{S}}$ and similarity w. Set each bin's count to zero.

3. Repeat the following steps to initialize learning for each classification algorithm.

   3.1. Create a histogram for estimating the distribution of all particle poses. Each bin is assigned a range for the similarity metric w. Set each bin's count to zero.

   3.2. Create a histogram for estimating the distribution of particles whose pose is near the ground truth $\vec{S} = \vec{S}^{True}$. Each bin is assigned a range for the similarity metric near w. Set each bin's count to zero.

4. Repeat the following steps to initialize learning for each feature extraction algorithm.

   4.1. Create a histogram for estimating the distribution of all particle poses. Each bin is assigned a range for the minimum distance metric $d_{\vec{f}}$. Set each bin's count to zero.

   4.2. Create a histogram for estimating the distribution of particles whose pose is near the ground truth $\vec{S} = \vec{S}^{True}$. Each bin is assigned a range for the minimum distance metric near $d_{\vec{f}}$. Set each bin's count to zero.

5. Repeat the following steps for each acquired image I_t, where I_t is a member of a set of randomly selected representative images of the target object $(I_0, \ldots, I_T)$.

   5.1. Compute the camera relative pose of the target object $\vec{S}^{True}$ using the current camera position and world-relative target object pose. This is the ground truth used for training.

5.2. Run each object recognition algorithm to create an estimate or estimates of the target pose $\vec{s}$ and associated similarity metric w using target image I_{EX} and unknown image I_t.

5.3. Run each low-level feature extraction algorithm to create a set of feature vectors $\{\vec{f}_0, \dots, \vec{f}_J\}$ using the unknown image I_t.

5.4. Generate a random collection of pose vectors $(\vec{S}_0, \dots, \vec{S}_N)$ distributed uniformly throughout the pose space defined by the camera's intrinsic parameters.

5.5. Repeat the following steps for each randomly generated pose vectors $\vec{S}$ a member of $(\vec{S}_0, \dots, \vec{S}_N)$.

5.5.1. Repeat the following steps for each object recognition algorithm.

5.5.1.1. Compute the Euclidean distance $d_{\vec{s}} = \|\vec{S} - \vec{s}\|$ between the particle pose $\vec{S}$ and the object recognition pose estimate $\vec{s}$.

5.5.1.2. Add one to the bin matching the values of $d_{\vec{s}}$ and w in the histogram counting all the pose estimates.

5.5.1.3. If the pose estimate $\vec{S}$ is near the ground truth pose $\vec{S}^{True}$, add one to the bin matching the values of $d_{\vec{s}}$ and w in the histogram counting all the pose estimates where $\vec{S} = \vec{S}^{True}$.

5.5.2. Repeat the following steps for each classification algorithm.

5.5.2.1. Create the affine transformation matrix $M_{\vec{S}}$ that corresponds to the pose $\vec{S}$.

5.5.2.2. Use the affine transformation matrix $M_{\vec{S}}$ to project the target image I_{EX} to the point $\vec{S}$.

5.5.2.3. Compare the projected target image $M_{\vec{S}} I_{EX}$ to the relevant portion of the image I_t using the classification algorithm to compute the similarity w.

5.5.2.4. Add one to the bin matching the value of w in the histogram counting all the pose estimates.

5.5.2.5. If the pose estimate $\vec{S}$ is near the ground truth pose $\vec{S}^{True}$, add one to the bin matching the value of w in the histogram counting all the pose estimates where $\vec{S} = \vec{S}^{True}$.

5.5.3. Repeat the following steps for each low-level feature extraction algorithm.

5.5.3.1. Compute the Euclidean distance between the particle's pose $\vec{S}$ and each feature vector $\vec{f}$ to find the feature vector closest to the particle's pose.

$$d_{\vec{f}} = \min_{j=0}^{J}(\|\vec{f}_j - \vec{S}\|)$$

5.5.3.2. Add one to the bin matching the minimum value $d_{\vec{f}}$ in the histogram counting all the pose estimates.

5.5.3.3. If the pose estimate $\vec{S}$ is near the ground truth pose $\vec{S}^{True}$, add one to the bin matching the minimum value of $d_{\vec{f}}$ in the histogram counting all the pose estimates where $\vec{S} = \vec{S}^{True}$.

6. Estimate $P(\vec{S} = \vec{S}^{True})$ by dividing the total number of random pose samples $\vec{S}$ by the total number of random pose samples near the ground truth $\vec{S} = \vec{S}^{True}$.

7. For each object recognition algorithm

   7.1. Estimate $P(d, w)$, using the histogram counting all the poses $\vec{S}$.

   7.2. Estimate $P(d, w|\vec{S} = \vec{S}^{True})$, using the histogram counting all poses near the ground truth poses $\vec{S} = \vec{S}^{True}$.

   7.3. Build the recognition-algorithm-specific look-up table of $\hat{P}(\vec{S} = \vec{S}^{True}|d, w)$ using Bayes' Theorem

$$\hat{P}(\vec{S} = \vec{S}^{True}|d, w) = \frac{\hat{P}(d, w|\vec{S} = \vec{S}^{True})\hat{P}(\vec{S}^{True})}{\hat{P}(\vec{S} = \vec{S}^{True})}$$

8. For each classification algorithm

   8.1. For each bin in the histogram counting all the poses $\vec{S}$, use the counts to estimate each $\hat{P}(w)$.

   8.2. For each bin in the histogram counting all poses near the ground truth pose $\vec{S} = \vec{S}^{True}$, use the counts to estimate each $\hat{P}(w|S = S^{True})$.

   8.3. Build the classification-algorithm-specific look-up table of $\hat{P}(\vec{S} = \vec{S}^{True}|w)$ using Bayes' Theorem

$$\hat{P}(\vec{S} = \vec{S}^{True}|w) = \frac{\hat{P}(w|\vec{S} = \vec{S}^{True})\hat{P}(\vec{S}^{True})}{\hat{P}(\vec{S} = \vec{S}^{True})}$$

9. For each low-level feature algorithm

9.1. For each bin in the histogram counting all the poses $\vec{S}$, use the counts to estimate each $\hat{P}(d_{min})$.

9.2. For each bin in the histogram counting all poses near the ground truth pose $\vec{S} = \vec{S}^{True}$, use the counts to estimate each $\hat{P}(d_{min}|S = S^{True})$.

9.3. Build the classification-algorithm-specific look-up table of $\hat{P}(\vec{S} = \vec{S}^{True}|d_{min})$ using Bayes' Theorem

$$\hat{P}(\vec{S} = \vec{S}^{True}|d_{min}) = \frac{\hat{P}(d_{min}|\vec{S} = \vec{S}^{True})\hat{P}(\vec{S}^{True})}{\hat{P}(\vec{S} = \vec{S}^{True})}$$

3.3.2 Object Recognition using the Extended Condensation Filter

Note: this step-by-step description uses the product rule

$$\hat{P}(S = S^{True}|w_0, \ldots, w_N) = \frac{\hat{P}(S = S^{True}) \prod_{i=1}^{N} \hat{P}(w_i|S = S^{True})}{\prod_{i=1}^{N} \hat{P}(w_i)}$$

to combine the recognition, classification, and low-level feature information sources.

1. Acquire target model, including the example target image I_{EX}, the target parameters (range to target), and the sensor parameters (field of view, resolution).

2. Initialize the particle pose vectors $\{\vec{S}_0, \ldots, \vec{S}_N\}$ by distributing them randomly throughout the pose space defined by the camera's intrinsic parameters.

3. Repeat the following steps for each acquired image I_t, where I_t is a member of the sequence of images $\{I_0, \ldots, I_T\}$.

   3.1. Run each object recognition algorithm to create an estimate or estimates of the target pose $\vec{s}$ and associated similarity metric w using target image I_{EX} and unknown image I_t.

   3.2. Run each low-level feature extraction algorithm to create a set of feature vectors $\{\vec{f}_0, \ldots, \vec{f}_J\}$ using the unknown image I_t.

   3.3. Repeat the following steps for each particle in the Condensation Filter population $\{\vec{S}_0, \ldots, \vec{S}_N\}$, where a single particle pose is $\vec{S}$.

      3.3.1. Set the particle's weight $w_{\vec{S}}$ to the value $\hat{P}(S = S^{True})$ estimated during the learning phase.

3.3.2. Repeat the following steps for each object recognition algorithm.

3.3.2.1. Compute the Euclidean distance $d_{\vec{s}} = \|\vec{S} - \vec{s}\|$ between the particle pose $\vec{S}$ and the object recognition pose estimate $\vec{s}$.

3.3.2.2. Retrieve the probability estimates $\hat{P}(d, w|S = S^{True})$ and $\hat{P}(d, w)$ from this recognition algorithm's look-up table using $d_{\vec{s}}$ and w as indices to the look-up table.

3.3.2.3. Implement the product rule form of Bayes' Theorem by multiplying the particle's weight by the ratio

$$\frac{\hat{P}(d_{\vec{s}}, w|S = S^{True})}{\hat{P}(d_{\vec{s}}, w)}$$

3.3.3. Repeat the following steps for each classification algorithm.

3.3.3.1. Convert the particle's pose estimate $\vec{S}$ to an affine transformation matrix $M_{\vec{S}}$.

3.3.3.2. Project the target image I_{EX} to the pose $\vec{S}$ using the transformation matrix $M_{\vec{S}}$ to create the new image $M_{\vec{S}}I_{EX}$.

3.3.3.3. Use the classification algorithm to compute the similarity w between the transformed image $M_{\vec{S}}I_{EX}$ and the unknown image I_t.

3.3.3.4. Retrieve the probability estimates $\hat{P}(w|S = S^{True})$ and $\hat{P}(w)$ from this recognition algorithm's look-up table using w as an index to the look-up table.

3.3.3.5. Multiply the particle's weight by the ratio

$$\frac{\hat{P}(w|S = S^{True})}{\hat{P}(w)}$$

3.3.4. Repeat the following steps for each low-level feature extraction algorithm.

3.3.4.1. Compute the Euclidean distance between the particle pose $\vec{S}$ and each feature vector $\vec{f}$ to find the closest feature vector to particle's pose.

3.3.4.2. Retrieve the probability estimates $\hat{P}(d_{\vec{f}}|S = S^{True})$ and $\hat{P}(d)$ from this recognition algorithm's look-up table using $d_{\vec{f}}$ and w as indices to the look-up table.

3.3.4.3. Multiply the particle's weight by the Bayesian ratio

$$\frac{\hat{P}(d|S = S^{True})}{\hat{P}(d)}$$

3.4. Create a new filter population by randomly selecting particles from the current filter population. Weight the random selection using the ratio of each particle's weight to the sum of all particle weights

$$P(reproduction) = \frac{P(S = S^{True})}{\sum_{j=0}^{J} P(S_j = S^{True}|\{w_0, \ldots, w_N\})}$$

3.5. Replace the current filter population with the new filter population.

3.6. Compute the camera-relative stochastic motion vectors $\vec{M}_{cm}$ and $\vec{M}_{tm}$, using the stochastic world-relative camera and target motion models.

3.7. Add the camera-relative stochastic motion vectors $\vec{M}_{cm}$ and $\vec{M}_{tm}$ to each particle in the filter population.

$$\vec{S}_i = \vec{S}_i + \vec{M}_{cm} + \vec{M}_{tm}$$

3.8. Test the single Gaussian assumption to extract a target pose estimate

3.8.1. Use bottom-up hierarchical clustering to combine the entire filter population. Compute the maximum value of the ratio

$$\frac{||\mu_1 - \mu_2||}{\sigma_1 + \sigma_2} \tag{3.28}$$

during the last two levels of the clustering process. The maximum ratio indicates the statistical distance between the clusters contained in the filter population — the greater the distance the less likely a single cluster accurately represents the entire population.

3.8.2. If the tests indicate a single Gaussian mode, use the mean pose $\mu_{\vec{S}}$ as the target pose estimate and the standard deviation $\sigma_{\vec{S}}$ as an estimate of the confidence associated with the target object pose estimate.

3.4 Condensation Filter Example

Two simple examples are provided in this section to help bridge the gap between the theoretical models and pseudo-code in this chapter and the complex filter implementation in the next chapter. The full implementation of the Extended Condensation Filter, described in the next chapter, uses multiple recognition, classification, and low-level feature extraction algorithms, resulting in a complex system. The complexity is sufficient to defy closed-form analysis and require empirical testing.

This section provides two examples simple enough to explain step-by-step. The first example demonstrates the operation of the filter using a single object recognition algorithm as input. The second example adds a second object recognition algorithms that provides distinctly different pose estimates. The examples illustrate the steps of learning the conditional probability, weighting particles, and reproducing particles.

In addition to providing a simple illustration of the filter's operation, the examples highlight a contradiction. The statistical filter model indicates the filter population should represent the conditional target pose probability. However, the second example indicates the population converges to a single statistically similar cluster of points, even when two equally good estimates are present. The precise definition of statistically similar is provided in Section 4.2.5. This apparent contradiction leads to further discussion in the final section of this chapter.

3.4.1 Single Expert

This first example implements a very simple one-dimensional Condensation Filter where one object recognition algorithm provides one target pose estimate s_t per filter iteration, t is the time or iteration index, and the first iteration is 1. The filter contains 10 particles and each particle i contains the pose S_i and a weight which is the probability estimate $\hat{P}(S_i = S^{True} \mid \|s_t - S_i\|)$. The term $\|s_t - S_i\|$ represents the distance between the current object recognition algorithm estimate s_t and the particle's pose S_i. For convenience, The symbol d_i represents the distance $\|s_t - S_i\|$. Note, the vector notation is dropped in this example to indicate this problem is purely one-dimensional.

3.4.1.1 Learning

Before processing an estimate from the object recognition algorithm, the filter must learn the conditional probability estimate. This relationship estimates the probability an arbitrary particle pose S_i is near the true pose S^{True}, given the distance d_i. Near the true target pose must be used, rather than the actual pose, because the probability of exactly hitting the real valued true target pose is 0. For this example, $S_i = S^{True}$ is implemented as $\|S_i - S^{True}\| < 0.05$, but the shorter notation $S_i = S^{True}$ is used for convenience.

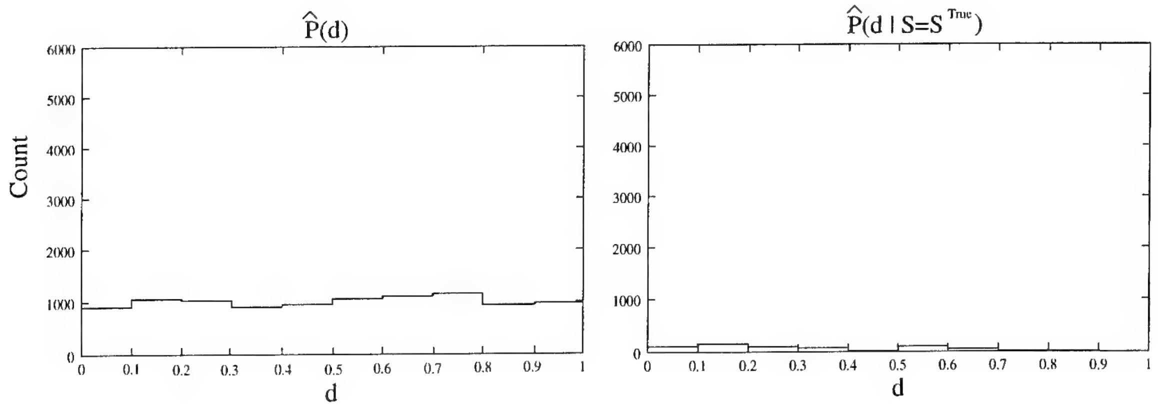


Figure 3.11: Conditional probability histograms after sampling the entire pose space.

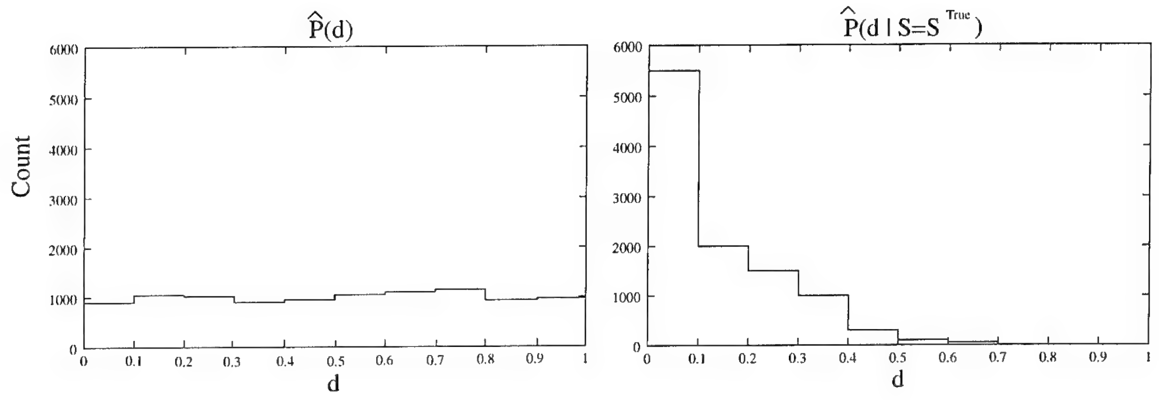


Figure 3.12: Conditional probability histograms after sampling just the ground truth portion of the pose space.

To learn a model of the conditional probability $P(S = S^{True}|d)$, iterations of random sampling are used to populate two histograms. Once populated, the first histogram will be used to estimate $P(d)$ and the second histogram will be used to estimate $P(d|S = S^{True})$. For each iteration of the learning algorithm, the following steps are performed

1. The target object is randomly placed at the pose S^{True}.
2. The object recognition algorithm generates one target pose estimate s_t in response to the scene containing the target object.
3. Ten random pose estimates $\{S_1, \ldots, S_{10}\}$ are generated uniformly in the pose space
4. The distance d_i is computed for each random pose estimate S_i, where $d_i = \|S_i - s_t\|$.
5. Every d_i is placed into the first histogram, shown on the left side of Figure 3.11, used to estimate $P(d)$.
6. Every distance d_i, where the pose estimate S_i is within the bounds of the true pose estimate $S_i = S^{True}$, is placed into the second histogram, shown on the right side of Figure 3.11, used to estimates $P(d|S = S^{True})$.

As shown in Figure 3.11, after 1,000 iterations of these steps, the first histogram will contain 10,000 samples. However, the pose space includes the real values from -1 to +1 and the true target pose is defined over an interval 0.1 units wide. Therefore, $P(S = S^{True})$ is 0.05 and the second histogram is only expected to contain about 500 of the 10,000 total samples.

To increase the number of samples populating the second histogram, learning is run again, except the 10 pose estimates S_i, generated in the third step, are randomly generated within the bounds of the true target pose, $S^{True} - 0.05$ to $S^{True} + 0.05$. After the second phase of learning, the first histogram contains only the original 10,000 distance values and the second histogram is expected to contain about 10,500 values. Figure 3.12 shows the histograms after adding 10,000 additional random pose estimates.

3.4.1.2 Filtering

Now that the model of the conditional probability $\hat{P}(S = S^{True}|d)$ is available, the filter is ready to operate. Before the first step of the first iteration, the filter is initialized by

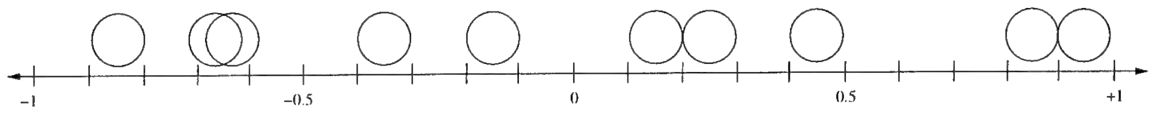


Figure 3.13: Initialized filter prior to iteration one.

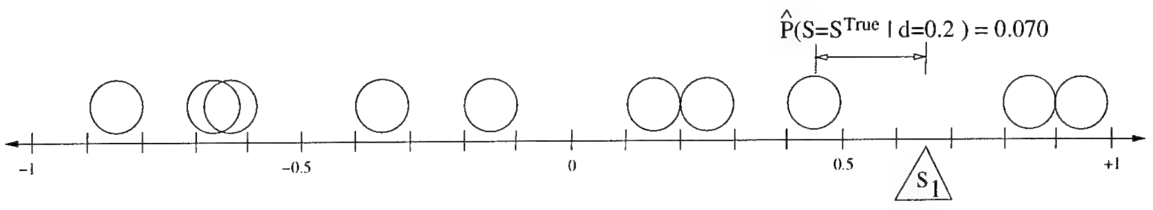


Figure 3.14: Step One - Measurements. Computing the weight $P(S = S^{True}|d)$ for one particle using the distance between the particle and the estimate s_1 is shown.

randomly distributing the population of ten particles $(S_1, \ldots, S_{10})$ uniformly throughout the pose space, as shown in Figure 3.13.

The first step of the first iteration is running the object recognition algorithm to generate an object pose estimate s_1. The estimate s_1 is shown in Figure 3.14.

Next, the distance d_i between the pose estimate s_1 and each particle $(S_1, \ldots, S_{10})$ is computed and used to look-up the corresponding probability estimate from the tables created in the previous section. Using Bayes' rule, the weight for a single particle is

$$\hat{P}(S_i = S^{True}|d_i) = \hat{P}(S = S^{True})\frac{\hat{P}(d_i|S = S^{True})}{P(d_i)} \tag{3.29}$$

For example, the weight for the particle compared to the estimate in Figure 3.14 is

$$\hat{P}(S_i = S^{True}|d_i = 0.2) = (0.05)(\frac{1,500/10,500}{1,020/10,000}) = 0.070$$

The distance and corresponding probability estimate for all the particles are computed in

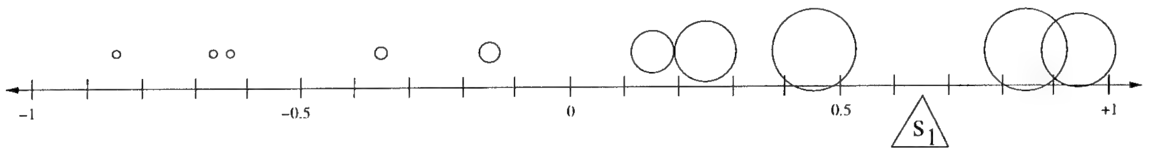


Figure 3.15: Step Two - Weighted Particles. Each particle's diameter is proportional to the log of its weight.

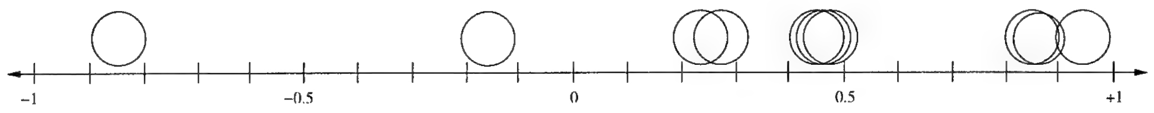


Figure 3.16: Step Two - Reproduction

a similar manner. Figure 3.15 shows the relative weights of the particles after the weighting step.

In the third step, the next generation of filter particles are created using weighted random reproduction of the current generation, where the probability of reproduction for each particle is the particle's weight divided by the total weight of the population.

When the next generation is produced, an uncertainty model is also applied to slightly disperse the particles. This uncertainty may be included in a model of object or camera motion, but for this static example a simple Gaussian is used to disperse the particles away from their positions in the previous population. Using statistical filter terminology, this random dispersion of particles represents uncertainty. Using Evolution Strategy terminology, this dispersion of particles is mutation. The second generation is shown in Figure 3.16. As you can see, the particles have begun to condense toward the object recognition algorithm estimate. However, due to the Gaussian uncertainty function, none of the particles occupies the exact same position in the second generation that it did in the first generation.

For each subsequent iteration, the process of obtaining a target pose estimate from the object recognition algorithm, measuring the distance between the estimate and each particle's pose, weighting each particle using the learned conditional probability, random reproduction of a new generation of particles, and adding the uncertainty is repeated. Given additional iterations and accurate pose estimates, the particles will condense toward the true target pose.

3.4.2 Two Experts

To extend the first example and illustrate an interesting issue, assume a second recognition algorithm is added to the system. For the example, assume the second expert is sensitive to a very different property of the image. For example, the first expert might use edge features and detect an object of similar shape on the right side of the image. The second expert might use color features and detect an object of similar color on the left side of the image.

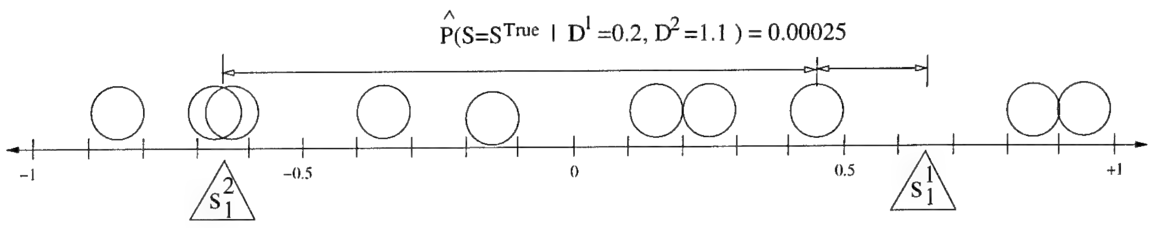


Figure 3.17: Step One - Two Measurements

For simplicity, assume the conditional probability models for both experts are identical to the model used in the first example.

Before the first iteration, the population of ten particles must be randomly distributed in the pose space. For simplicity, this example starts with the same random initialization of particles shown in Figure 3.13.

The first step of the first iteration obtains the estimates s_1^1 and s_1^2 from the two experts — the estimates are shown in Figure 3.17.

Next, each particle must be compared to both estimates s_1^1 and s_1^2. When two estimates are available, each particle is weighted using an estimate of the probability conditioned on both estimates $\hat{P}(S_i = S^{True}|d^1, d^2)$. The two conditions are combined using the Bayesian product rule

$$\hat{P}(S = S^{True}|d^1, d^2) = P(S = S^{True})\frac{P(d^1|S = S^{True})}{P(d^1)}\frac{P(d^2|S = S^{True})}{P(d^2)} \tag{3.30}$$

For example, the weight for the particle compared to the two estimates in Figure 3.17 is

$$\hat{P}(S_i = S^{True}|d^1 = 0.2, d^2 = 1.1) = (0.05)(\frac{1,500/10,500}{1,020/10,000})(\frac{1/10,500}{525/10,000}) = 2.5E-4$$

Once each particle is assigned a weight, the next generation is produced. Given two estimates with identical conditional probability models, the population will initially condense toward the two equally good estimates. After a large number of iterations, the population should resemble the underlying conditional probability. However, running this process described in this section many times consistently leads to a single cluster of points centered on one of the two expert estimates. The probability of converging to the left or right estimate appears to be about equal.

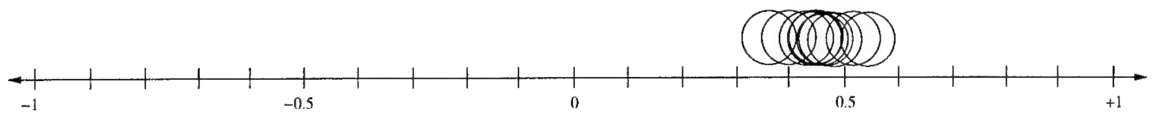


Figure 3.18: Filter state after 50 iterations. The population has converged to represent a single pose estimate.

3.5 What Does the Filter Population Represent?

Both Gordon and Blake indicate the Condensation Filter population represents the underlying conditional probability for the state space. According to Gordon

> The bootstrap filter is an approach where random samples are used to represent the target posterior probabilities [Gor97].

According to Blake

> The Condensation algorithm uses "factored sampling", previously applied to the interpretation of static images, in which the probability of possible interpretations is represented by a randomly generated set [IB98].

Applying these descriptions to the example in the previous section using two experts, the filter population should *represent* the conditional probability defined by $\hat{P}(S = S^{True}|d^1, d^2)$. The key term *represents*, as defined by [IB98] and [Mac00], means the weighted density of particles per unit of pose space is approximately the same as the conditional probability function $P(S = S^{True}|d^1, d^2)$. However, this does not appear to be the case in the previous section's simple example. Two regions of the pose space, one around each observation, are equally probable. Therefore, one expects a roughly equal number of particles around each. This is not what is observed. Instead, after several iterations, one region contains no particles, while the other contains all the particles.

The literature provides two proofs that the Condensation Filter should represent the conditional probability. First, Blake provides an inductive proof. However, the proof requires the population of particles be independent and identically distributed for each iteration [Bla92]. As noted by MacCormick, the population of particles is not independent and identically distributed throughout the pose space after the first iteration, therefore the inductive proof is incorrect [Mac00]. In addition to showing a weakness in Blake's proof, MacCormick provides a proof using lemmas from DelMoral's work [DL00]. MacCormick's proof deserves further study.

MacCormick provides an inductive proof that the population of a Condensation Filter represents the conditional probability based on describing the Condensation Filter as three operations: multiplication by a function, applying dynamics, and resampling. These three steps correspond to weighting, prediction, and reproduction using the terminology in my dissertation. In MacCormick's dissertation, the validity of each operation is proven as a theorem. However, MacCormick's inductive proof requires these three operations be applied repetitively. When applied repetitively, one of these three operations (resampling) does not appear to produce the desired result. Briefly examining the first two theorems, then carefully examining the Resampling Theorem, illustrates the potential problem with the proof.

Multiplication Theorem Let $\{S\}$ represent $P(x)$, where $\{S\}'$ is obtained by multiplication by $h(x)$, where $h(x)$ is continuous, non-negative, and $h(x)P(x) \neq 0$, then $\{S\}'$ represents $h(x)P(x)$.

This theorem relies on the concept of approximate representations of continuous functions using weighted discrete representations. The continuous uniform distribution $P(x)$ shown in Figure 3.19(a) can be represented in different ways. As shown in Figure 3.19(b), a set of N equally weighted particles $\{S\}$ can be placed $1/N$ units apart over the interval from one to 10, and this provides a discrete representation of the uniform distribution. Alternatively, as shown in Figure 3.19(c), N particles can be placed with equal probability anywhere on the interval — this also represents a uniform distribution. As shown in 3.19(d), particles can also be weighted, and fewer particles with larger weights can be used in place of smaller particles with lower weights. Each of these representations, using weighted discrete integration, will approximate the continuous uniform distribution shown in the first plot.

Intuitively, the Multiplication Theorem is simply applying a weighting to a collection of particles and showing the weighted particles, if weighted integration is performed, will approximate the original distribution $P(x)$ times the function $h(x)$. MacCormick's proof of this theorem relies on a lemma provided by DelMoral [Mac00, DL00].

Dynamics Theorem If $\{S\}$ represents $P(x_{t-1})$ and $\{S\}'$ is obtained by applying the Markov dynamic model $P(x_t|x_{t-1})$, then $\{S\}'$ represents $P(x_t)$.

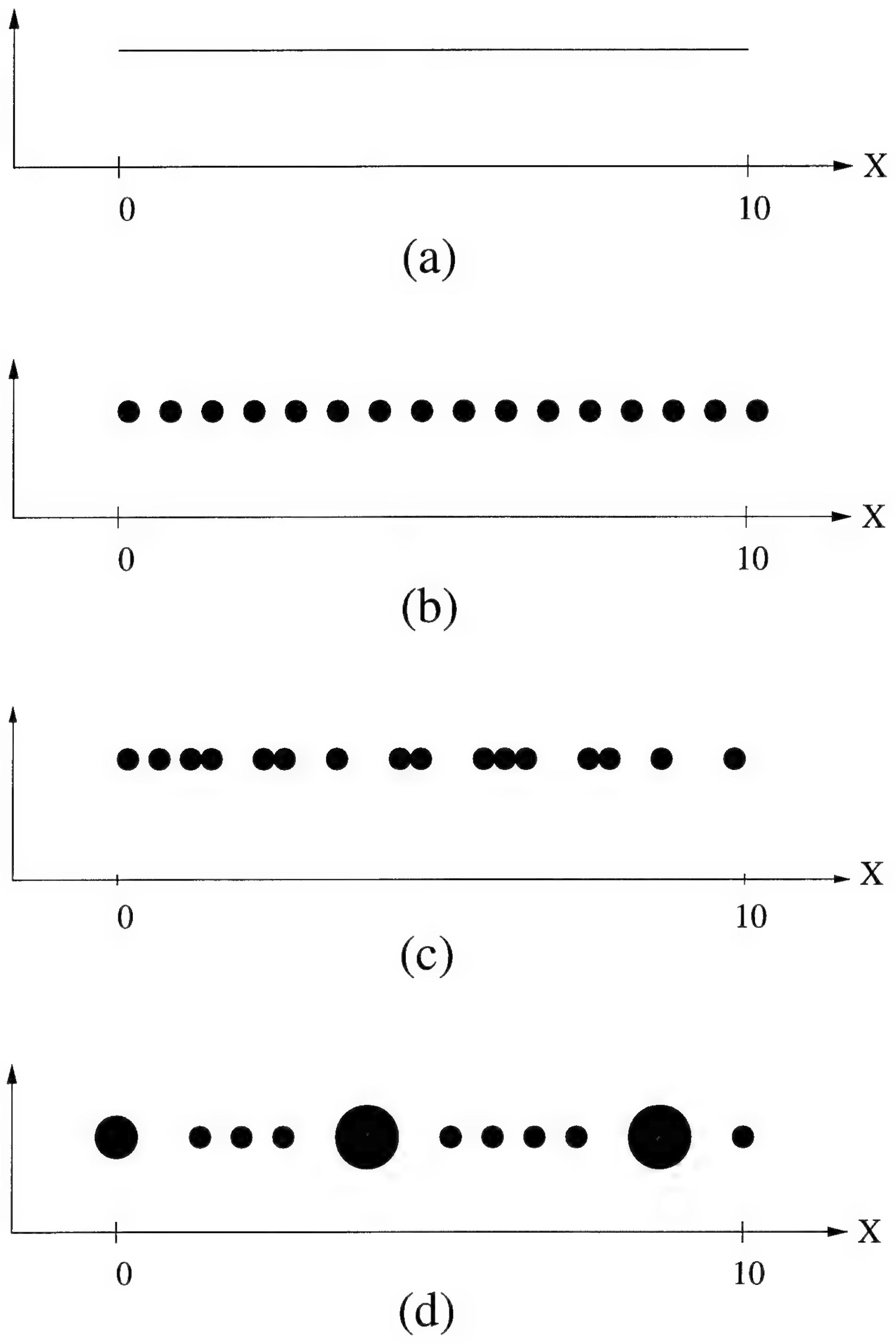


Figure 3.19: Three ways to represent the continuous uniform distribution $P(x)$ shown at the top. (b) using a deterministic uniform distribution of points. (c) using a random uniform distribution of points. (d) using a deterministic weighted uniform distribution of points. Using weighted discrete integration, all three of the particle sets approximate the continuous distribution.

The Dynamics Theorem provides a method of shifting the current particle population $\{S\}$, which currently represent the distribution $P(x_{t-1})$, in accordance with a Markov prediction of the next state $P(x_t|x_{t-1})$, to achieve a population of particles which predict the distribution $P(x_t)$ at the next state. This creates a population of particles that predict the next probability distribution according to the current distribution of particles and the motion model.

The third, and suspect theorem, is the Resampling Theorem.

Resampling Theorem If $\{S\}'$ is a weighted random sampling of $\{S\}$, and $\{S\}$ represents $P(x)$, then $\{S\}'$ represents $P(x)$.

Intuitively, the Resampling Theorem states that drawing N uniform random samples, with replacement, from a set of N samples representing a distribution $P(x)$ will create a new set which also represents the distribution $P(x)$. For one iteration, given a large population of samples, this is true. For example, given the three discrete sets of particles shown in Figure 3.19, 10 uniform random draws with replacement from the population is expected to produce a population approximating the uniform random distribution.

A problem with the Resampling Theorem appears when it is repetitively applied, as required for an inductive proof. The repetitive nature of an inductive proof requires the theorem hold even when applied many times, and an example shows at least one case where repeated application of the theorem does not produce the same distribution.

Consider the uniform random population of particles shown in Figure 3.19(c). For each draw from this population, the probability a particle will not be selected is

$$P_{lost} = \frac{N-1}{N} \tag{3.31}$$

where N is the population size. Since each draw from the previous generation is performed with replacement, they are independent events. Therefore, for the reproduction of the entire next generation, the probability of not being selected is

$$P^N_{lost} = \left(\frac{N-1}{N}\right)^N \tag{3.32}$$

Applying this equation to any of the discrete representations in Figure 3.19, all of which have 16 particles, the probability of not reproducing any single particle from the current generation into the next generation is

$$P^N_{lost}(N=16) = \left(\frac{15}{16}\right)^{16} = 0.356$$

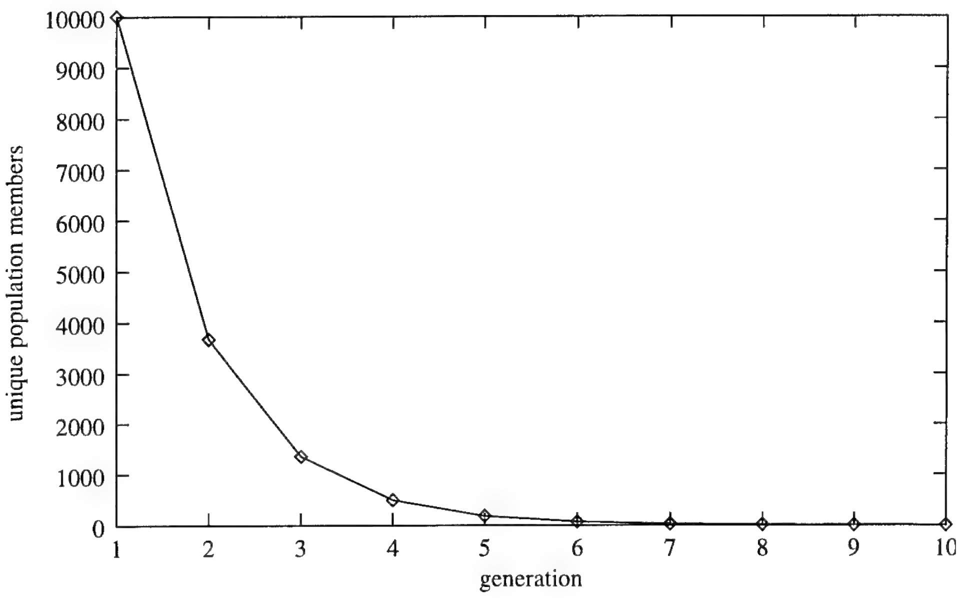


Figure 3.20: The expected number of unique population members in each generation given a population of 10,000 unique population members and a uniform distribution. Only the resampling operation is applied for each next generation.

Therefore, in expectation, 0.356 * 16 or 5.7 of the particles will not be used to create the next generation. For much larger populations, the percentage of particles not represented in the next generation is approximately the same. For example, for N = 10,000 particles, the percentage of particles not used for the next generation is

$$P_{lost}^{N}(N = 10,000) = \left(\frac{9,999}{10,000}\right)^{10,000} = 0.368$$

This shows, in expectation, each succeeding generation will fail to represent about 35% of the population in the previous generation. This means the population very rapidly contains N identical particles. Figure 3.20 shows an example which starts with a population of 10,000 unique population members. The population is expected to contain 10,000 identical particles in approximately 10 generations.

The exact use of the resampling theorem in an inductive proof is critical to the correctness of the proof. Unfortunately, MacCormick fails to address this problem in the following brief statement provided as the inductive proof.

> *Proof* This is a trivial induction using theorems [Multiplication], [Dynamics], and [Resampling]. The necessity for a Condensation diagram to be well formed is because of the assumption made in Theorem [Predict].

As a result, some questions remain about the claim that the Condensation Filter represents the conditional probability.

3.6 Research Questions

As explained by Cohen, formal empirical hypotheses focus on extremely narrow questions with yes or no answers [Coh95]. An empirical hypothesis is formulated, statistics are compiled for a representative data set, and the statistics either support or refute the hypothesis with some specified level of confidence. Empirical hypotheses are like a guessing game where only yes or no questions are allowed. The open-ended nature of research questions expand the possible responses but lead to greater ambiguity and need for interpretation. The Extended Condensation Filter described in this chapter suggests many research questions, but resource limitations only allow a few to be investigated. The following research questions appear to be the most interesting and provide the focus for the remainder of this dissertation.

1. How does the discrimination of the combined probability estimate compare to the discrimination of the probability estimate from each of the contributing experts?

2. How does the accuracy and precision of the Extended Condensation Filter's combined pose estimate compare to the accuracy of the pose estimate from each of the contributing experts?

3. Does the Extended Condensation Filter's learning algorithm generalize sufficiently to support operation in new portions of the problem domain?

4. Is the weight for each particle required to be a probability estimate or, as Pearl indicates [Pea88], is relative likelihood sufficient?

5. What are the convergence properties of the filter population? Specifically, how common is a multi-mode distribution of filter particles?

In the next chapter, a specific implementation of the Extended Condensation Filter, a set of test environments, and a set of formal empirically testable hypotheses will be designed to address these research questions.

3.7 Summary

This chapter has described the Extended Condensation Filter using two models, pseudocode, and two simple examples. Each model and the examples provide a different view of the Extended Condensation Filter. The statistical filter model illustrates the proportional low-pass proportional filter and predicts the population will converge to the conditional target pose probability. The search model illustrates the filter's ability to combine the diverse information provided by recognition, classification, and low-level feature data and predicts the population will converge to the maximum probability. The examples illustrate the filter implementation and a difference between theory and implementation.

The models in this chapter indicate the Extended Condensation Filter achieves the goals stated at the beginning of this chapter.

1. The Extended Condensation Filter does not rely on specific parametric representations. A simple look-up table of arbitrary resolution is used to model uncertainty.

2. The fusion of visual information is guided by estimates of $P(\vec{S} = \vec{S}^{True})$ to reduce the loss of information that occurs with discrete decision boundaries.

3. Visual information at different levels of abstraction is accommodated. Recognition pose estimates, classification metrics and low-level features are combined using the population of particle poses as a common reference.

The models in this chapter prompted research questions regarding the discrimination, accuracy, precision, generalization, sufficiency of likelihood, and convergence of the filter. Each of the models can be used to support answer to the research questions, but none of these models can be empirically tested. The next chapter describes the tools necessary to perform empirical testing, including a set of testable hypotheses, a set of test environments, and an implementation of the Extended Condensation Filter.

Chapter 4

Test

This chapter describes a specific implementation of the Extended Condensation Filter, a set of problem environments, and a set of hypotheses that prepare for the empirical testing reported in the next chapter. The implementation, problems, and hypotheses are highly interrelated, so I have elected to describe the problem environments first to provide a context for the other discussions. The three problem environments are Block World One, Block World Two, and Toy World. These environments provide synthetic, but realistic, examples of a broad class of search problems common to law enforcement, rescue, and military operations. The second section of this chapter describes a set of empirically testable hypotheses for each of the research questions posed at the end of the previous chapter. The yes or no answers for the testable hypotheses, provided in the next chapter, support answers to the more general research questions proposed in the previous chapter. The implementation is discussed third because the problem environments and hypotheses drive its design. The last section describes a second set of implementation specific research questions that complement the research questions.

4.1 Problem Environments

In the introduction to this dissertation I used a search problem to illustrate the benefits of fusing recognition, classification, and low-level feature extraction. The example involved a police car searching for a suspect vehicle. Searching for and tracking a target object while traversing a predefined path is a common problem in search and rescue, law enforcement, and military operations. Even some mail delivery systems, such as the one the Pentagon employs, could use such a system to find and navigate to the next mail drop. The common problem of identifying and tracking a target object while traversing a predefined path is the basis for all the synthetic problem environments used in this dissertation.

4.1.1 Simulation

All of the test imagery for this dissertation is produced using the photo-realistic Persistence of Vision Ray-tracer (POVRay). [4] Details about the configuration and control of POVRay are provided in Appendix B. Testing vision systems with synthetic imagery leaves open the question of how applicable the results are to physical problems. However, using synthetic imagery for the initial testing of a vision system has several significant advantages.

First, synthetic imagery provides a degree of insight simply not available at even the most capable physical test facility. Major test facilities, such as the Air Force Automatic Target Recognition Lab, the Air Force Flight Test Center, or the Western Range offer accurate sources of ground truth, such as differential GPS and cinetheodelite. They also provide high-bandwidth remote monitoring. However, even these multi-billion dollar national facilities cannot compare with the degree of scrutiny provided by a desktop computer using synthetic imagery. Using synthetic imagery, I know the location of every target and camera to 64-bits of double-precision and I can record any parameter at any resolution. No physical test facility can match this level of scrutiny.

Second, greater control is provided by synthetic imagery. To provide the controlled and repeatable test conditions required for empirical testing, laboratories must take extreme measures. Indoor laboratories require the precise calibration and positioning of cameras, target objects, backgrounds, and lighting. In outdoor test environments, replicating test conditions is nearly impossible due to factors such as natural lighting and weather. For synthetic imagery, repeatable test conditions are a given — the inherently deterministic nature of a digital computer ensures the same model produces the same image. When using synthetic imagery, I can repeat test conditions precisely or change one, and only one, test condition. No physical test facility can match this degree of control.

Third, physical cameras looking at physical objects do not guarantee a good simulation. A lab test using a camera and a ray-tracer using a mathematical model are both simulations,

[4]The Persistence of Vision Ray-tracer (POVRay) is available at www.povray.org. Version 3.1g was used for this dissertation.

unless the lab is the intended environment for the vision system. If, for example, a vision system is tested in a lab with painted surfaces and fluorescent lights, and the system will operate in a jungle, the bench test may provide more misleading results than a well-designed synthetic environment. The synthetic environment can easily include the broken light provided by a jungle canopy and the geometry and textures of jungle foliage. Identifying the relevant properties of the intended environment and including those in the test improves the fidelity of the simulation, whether the simulation is performed in a ray-tracer or on a lab bench. Unless the lab happens to be similar to the target environment, including these properties is usually easier in a synthetic environment.

To illustrate this point, consider one of the 1999 AAAI Competition entries that served hors d'oeuvres [M+99]. The robot relied on a color histogram algorithm to detect the presence of a human hand and count the number of hors d'oeuvres served. When the number of times a hand was detected equaled the initial number of hors d'oeuvres, the robot returned to its base to refill its tray. The robot worked well in lab tests and in the North Dakota Museum of Art, but failed during the AAAI competition. The walls of the room for the AAAI Competition resembled the color of human skin. The robot constantly detected the presence of a hand, immediately assumed all its hors d'oeuvres had been served, and returned to base without serving anything. The robot was thoroughly tested in two physical environments, but from the point of view of the AAAI competition, these environments were poor simulations. Texture mapping the competition walls into a rendered environment may have provided a superior simulation, at least so far as the color histogram algorithm was concerned.

During the 2001 Game Developers Conference, Marvin Minsky made the following comment in his presentation titled "Programs, Emotions and Common Sense."

> In the real world, how do you know your simulation contains all the important things? The answer is, tell me an important thing you've discovered and I'll make sure it gets into the simulation, and nobody every tells me one. Yes, the gears can have backlash and I'll bet that your best CAD program can allow for that. Yes, the floor might be slippery and I'll bet there are seven kinds of friction. But then you can have the robot go over these 100 times varying a few parameters and understand the problem. But what happens in physical

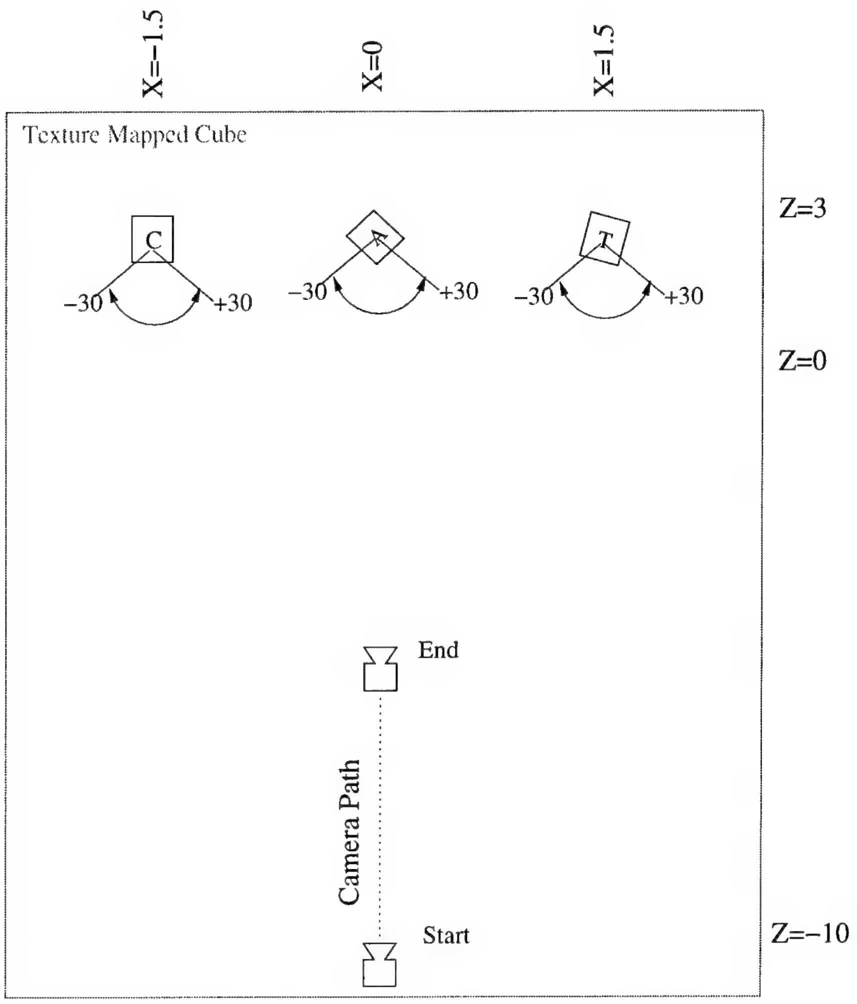


Figure 4.1: Block-World One Object Placement and Camera Motion

robotics, is you never get to do the same thing twice, there's no science. There's no replicable experiment... So, I think simulations are immensely important.

Synthetic testing is the proper first step when the goal is carefully controlled empirical study. The final step is testing on actual data from physical sensors in the system's target environment. However, as already suggested, controlled experiments using physical sensors, moving platforms, and physical targets is highly involved and beyond the scope of this dissertation. If this is not immediately obvious, again consider the billions of dollars the Federal Government spends establishing such testing facilities for processes of comparable complexity to that described in this dissertation.

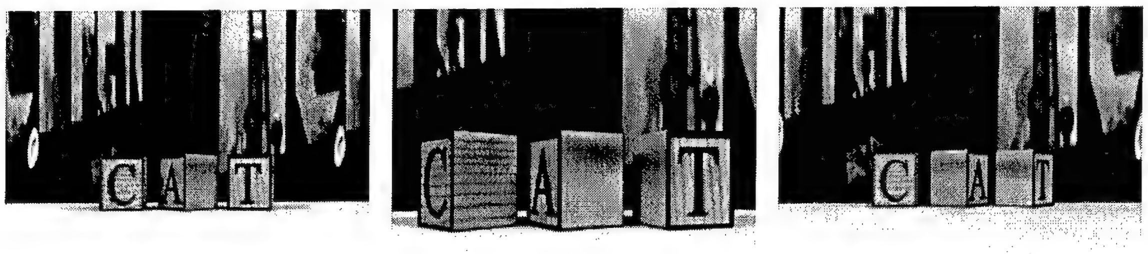


Figure 4.2: Typical images from the Block World One problem environment

4.1.2 General Description of the Problem Environments

The Block-World One, Block-World Two, and Toy World problem environments are inspired by the problem of searching for an object using a predefined search path. This problem is common in search and rescue, law enforcement, and military operations. Figure 4.1 shows the predefined path used in all three problem environments. The camera always starts at the location (x=0, y=0.5, z=-10) and moves forward 3.2 units in the $+z$ direction. The camera is always looking at the point (x=0, y=0.5, z=0). Three unoccluded target objects, a background image, and the floor are always visible. Typical images from the camera's point of view are shown in Figure 4.2.

Two cameras are simulated to support the stereo disparity algorithm that is part of the Extended Condensation Filter implementation. The cameras are mounted on a fixture so they are always at the same relative position. The left camera is always on the z-axis, while the right camera is always plus 0.25 units along the x-axis. For example, if the left camera is located at (x=0, y=0.5, z=-10), then the right camera is located at (x=0.25, y=0.5, z=-10).

A small amount of ambient light and two directional area light sources are used in all the problem environments. One area light source is centered at (-10, 10, -10) and the other is centered at (10, 10, -10). Both light sources are pointed at the origin. To improve realism, each area light source is modeled as a 5x5 array of adaptively sampled point light sources with spherical power drop-off. In practice, area light sources create soft complex shadows and cause the apparent color of an object to change as it moves around the environment.

For example, targets further from the light sources appear darker and objects may create shadows on other objects.

For each problem environment, 60 movies of 33 frames each were rendered. There are actually 66 images in each movie because two cameras are modeled in each frame. Each of the movies provide three tracking problems because each movie contains three objects that can act as either the true target or a false target. For example, in the Block World environments, the "A" block can be tracked in movie one, then the "C" block can be tracked in movie one, then the "T" block can be tracked in movie one. Using each movie to create three tracking problems provides 180 tracking problems in each of the three problem environments. Fifty of the movies, or 150 tracking problems, in each environment are used for training and some limited testing, while ten of the movies in each problem environment are withheld and used only for testing.

Table 4.1: The limits for Block World One randomly generated object poses. The angle ψ is the world-relative rotation of the target object, where zero is parallel to the negative z-axis. The * indicates the z coordinates of the "A" and "T" blocks must be the same as the "C" block. The randomly generated pose for each of these targets becomes the ground truth target object pose $\vec{S}^{True}$.

Target Object	Pose Parameter	Min	Max
"C" Block	X	-1.5	-1.5
"C" Block	Z	0	3
"C" Block	ψ	-60	60
"A" Block	X	0	0
"A" Block	Z	*	*
"A" Block	ψ	-60	60
"T" Block	X	1.5	1.5
"T" Block	Z	*	*
"T" Block	ψ	-60	60

4.1.3 Block World One Description

Block World One contains the three alphabet blocks "A", "C", and "T" shown in Figure 4.2. To support random placement of the target objects while preventing occlusion, the

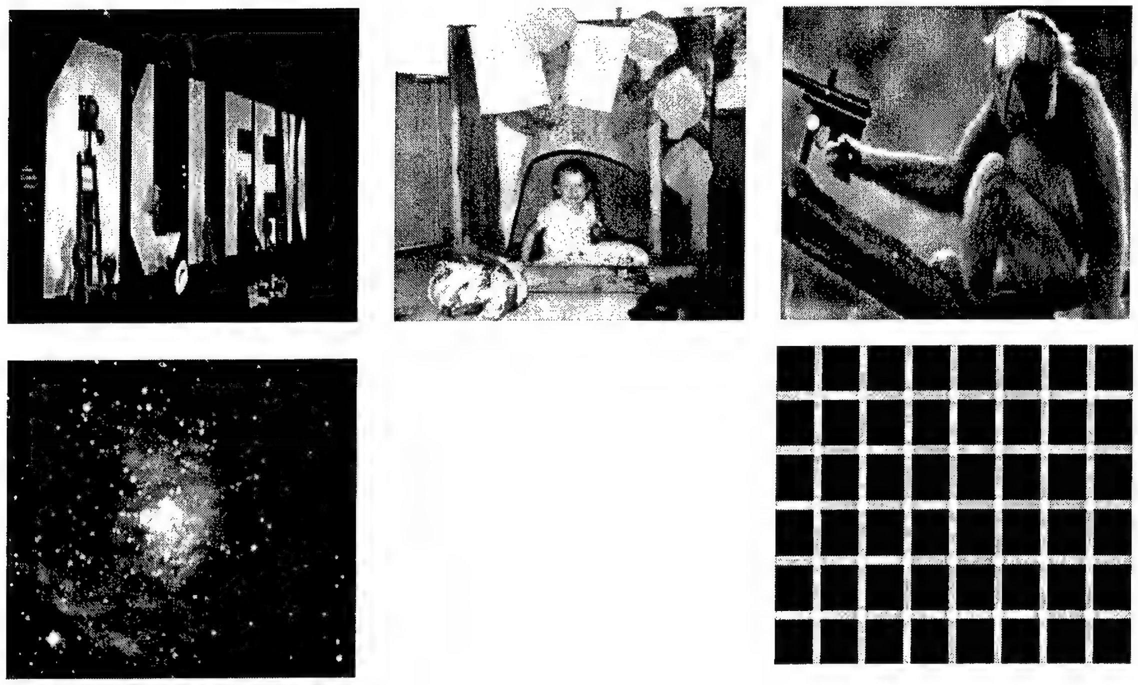


Figure 4.3: Backgrounds used for Block World Two. Only the first background (upper left) was used for Block World One.

left block "C" is always at x=-1.5, the middle block "A" is always at x=0, and the right block "T" is always at x=1.5. For each movie, all three blocks are placed at the same randomly generated z position between z=0 and z=3. In addition to translation, each of the blocks is randomly and independently rotated by plus or minus 60 degrees. These pose constraints prevent target object occlusion and ensure the target objects are always visible as the camera travels its predefined path. The randomly generated pose for each of these targets is the ground truth target object pose $\vec{S}^{True}$. The range of world-relative target object poses in Block World One is shown in Table 4.1.

4.1.4 Block World Two Description

After running experiments in Block World One, I grew concerned that the filter might learn the background appearance or some property of the target object's location rather than learning to recognize the target. Block World Two is designed to provide greater variability in the background and relative locations of the target objects.

Three changes were made to Block World One to create Block World Two. First, each target object is assigned an independently generated z location rather than assigning all the

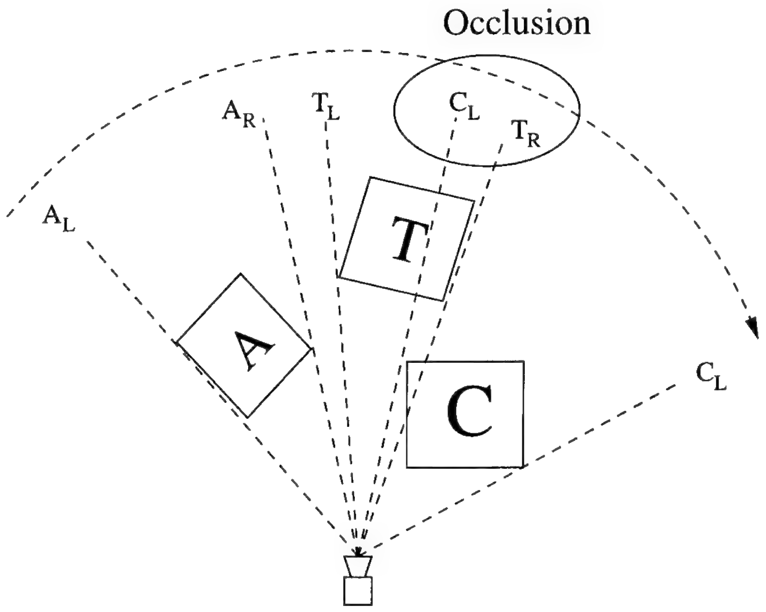


Figure 4.4: Testing for Occlusion. If camera-relative viewing angles for any of the objects overlap, as indicated with the middle "T" and right object "C", the set of target object poses is discarded.

target objects the same randomly generated z location. Second, each block in Block World Two is assigned a random x value rather than a constant x value. As a result, each target block may appear on the left, middle, or right side of the image. Third, the background is randomly selected from the five backgrounds shown in Figure 4.3.

To prevent the occlusion problems created by the additional variability of Block World Two, the camera-relative view angle for the extreme left and right edge of each object are compared, as shown in Figure 4.4. If, for example, the viewing angle to the extreme right side of the middle target object is greater than the viewing angle to the extreme left side of the right target object, either the left or middle object must be occluded. This random combination is discarded and another is generated. Only random combinations with no occlusion are used to render movies. The range of Block World Two target object poses is shown in Table 4.2. Typical images from Block World Two movies are shown in Figure 4.5.

4.1.5 Toy World

Toy World is very similar to Block World Two with one exception — the highly rectilinear alphabet blocks are replaced with curved objects — a stuffed cat, a pig riding on a cart, and Rubik's cube. These objects are shown in Figure 4.6.

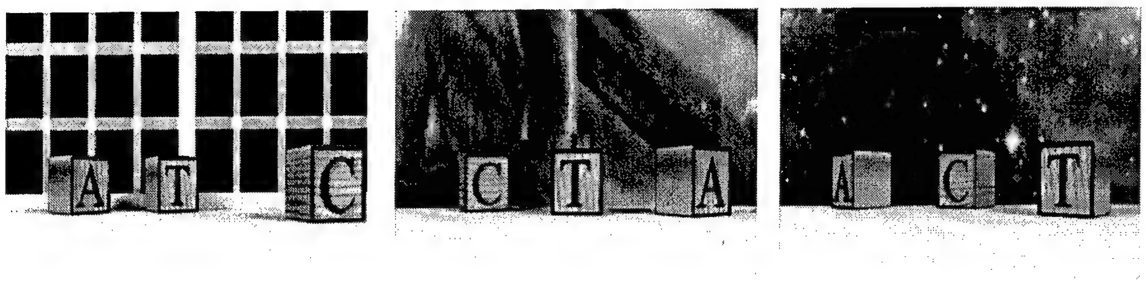


Figure 4.5: Typical images from the Block World Two problem environment.

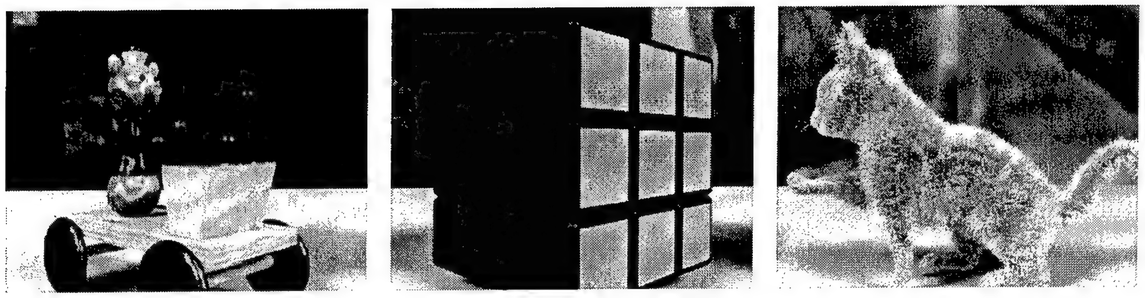


Figure 4.6: Toy World target templates (pig, Rubik's cube, cat)

Table 4.2: The limits for Block World Two and Toy World randomly generated object poses. Note, z is generated independently for each object. All coordinates are world-relative.

Target Object	Pose Parameter	Min	Max
"C" Block	X	-2.5	2.5
"C" Block	Z	0	3
"C" Block	ψ	-60	60
"A" Block	X	-2.5	2.5
"A" Block	Z	0	3
"A" Block	ψ	-60	60
"T" Block	X	-2.5	2.5
"T" Block	Z	0	3
"T" Block	ψ	-60	60

As described later, in the implementation section, the Extended Condensation Filter implementation uses a simple two-dimensional target object model. There is nothing in the representational requirements of the Extended Condensation Filter that precludes the use of three-dimensional target objects models. Others have used standard Condensation Filters for tracking three-dimensional objects. It may be argued that tracking given a fully three-dimensional object model is easier than the problem considered in this thesis — a three-dimensional model provides more complete information about the target object. However, this hypothesis is not tested here.

The Toy World problem environment is designed to investigate the problems created by the implementation's use of a single two-dimensional polygon to model three-dimensional objects projected into a six-dimensional world. Intuitively, the pig and the cat should be more challenging for the two-dimensional target projection used by the classification algorithms. Projecting the pig or cat images onto a single polygon, then rotating the polygon is not an accurate representation of the appearance of the three-dimensional object when rotated. The Rubik's cube target used in Toy World could be accurately represented by a two-dimensional polygon. However, as shown in Figure 4.6, the target template for the Rubik's cube has been made from an edge-on view, which is not a flat surface well modeled by a single polygon.

To provide a controlled experiment, the other aspects of Toy World, such as lighting, camera motion, and the range of object poses, are identical to Block World Two. Table 4.2 shows the range of object pose parameters for both Block World Two and Toy World. Changing only one property of the problem domain permits the attribution of statistically different results to the change — a basic tenant of the controlled experiment. If more than one property were changed and different results were observed, additional experiments would be required to determine which change or combination of changes was the cause.

4.2 Empirically Testable Hypotheses

The research questions in Chapter 3 are too general to test directly. However, answers can be inferred from testable hypotheses based on the research questions. Each of the following sections restates one of the research questions, describes my expectations regarding the question, states a testable hypothesis based on these expectations, and describes the data and analysis required to test the hypothesis.

4.2.1 Research Question: Discrimination

Research Question How does the discrimination of the combined probability estimate compare to the discrimination of the probability estimate from each of the contributing experts?

To perform target recognition, the experts must provide the extended condensation filter with probability estimates that discriminate between instances of the target pose $\vec{S} = \vec{S}^{True}$ and non-target pose $\vec{S} \neq \vec{S}^{True}$. This probability estimate is used by the filter to weight each particle, but it is independent of the filter's predict-weight-reproduce cycle. In other words, discrimination can be evaluated independently of the filter's weight-predict-reproduce cycle. Shortly, I will show the discrimination must be evaluated independently of the filter's operation.

The probability estimates used to weight the particles represent a simple two-class classification problem. The area under the ROC curve (AUC), described in Chapter 3, is a well established statistic for evaluating the discrimination of a system for two-class problems [Swe64]. To claim improved classification, the discrimination provided by the particle weights using all the experts should be greater than the discrimination using each contributing expert individually.

Hypothesis The AUC of the combined probability estimate is less than the AUC for the probability estimate using each contributing expert independently.

Each expert provides an estimate of $\hat{P}(\vec{S} = \vec{S}^{True}|w_i)$, where w_i is one of the expert similarity measures described in the next section on implementation. For example, the render-match algorithm provides the correlation w_{RM} between the projected target image MI_{EX} and the unknown image I_t, where M is an affine transformation projecting the target model image to the particle pose $\vec{S}$. The value w_{RM} is used to look up the value $P(\vec{S} = \vec{S}^{True}|w_{RM})$. This probability estimate is a discriminator, where higher values should indicate greater similarity between the target and unknown image. The discriminator $P(\vec{S} = \vec{S}^{True}|w_i)$ is evaluated using the AUC statistic.

4.2.2 Research Question: Accuracy and Precision

Research Question How does the accuracy and precision of the Extended Condensation Filter's combined pose estimate compare to the accuracy of the pose estimate from each of the contributing experts?

In my Extended Condensation Filter implementation, a pose estimate is three-dimensional — two translation dimensions x and z, and one rotation dimension θ. The first part of this section describes the circular error probable (CEP) statistic commonly used to evaluate the accuracy and precision of pose estimates. Unfortunately, the basic CEP statistic can provide misleading results when the error in different dimensions is grossly different. The basic CEP statistic can also provide misleading results when the dimensions of estimates are dissimilar, such as rotation and translation. The second part of this section describes the scaled CEP statistic, which statistically normalizes the different dimensions of the estimate error. The scaled CEP statistic will be used in the next chapter to present accuracy and precision test results.

4.2.2.1 Circular Error Probable (CEP)

If the filter population consistently resembled a Gaussian, the mean and standard deviation of the error could be used to represent the accuracy and precision of the filter's estimate.

Unfortunately, as shown in Figure 4.8, the filter population may contain multiple well-separated clusters of points. Where a well-separated cluster is defined later in Section 4.2.5. When multiple clusters are present, the mean value may represent a pose that the filter does not support.

The CEP statistic has several useful properties. First, CEP provides a single number that measures the error contained in multi-dimensional estimates. Second, CEP provides a single number that decreases in proportion to increasing accuracy and increasing precision. Third, it requires no assumption about the distribution of the estimate error. Each of these properties are addressed in this section.

Even for multi-dimensional estimates, a single distance measure is used to compute the CEP. For example, the Euclidean distance

$$\sqrt{(S_x^{True} - s_x)^2 + (S_z^{True} - s_z)^2} \tag{4.1}$$

can be used, where S_x^{True} and S_z^{True} are the x and z components of the ground truth pose $\vec{S}^{True}$, and s_x and s_z are the x and z components of a particle's pose estimate $\vec{s}$. Additional dimensions, such as the vertical translation y, can easily be incorporated.

Figure 4.7 illustrates the use of the CEP statistic. The diamonds in the figure show the pose estimate error for each particle. The radius of the large circle is the CEP and the circle is centered on the ground truth pose $\vec{S}$. If more dimensions are used, the CEP becomes the radius of a sphere or hyper-sphere. Regardless of the number of dimensions, the CEP provides a single statistic where half the estimates fall inside the radius.

CEP also provides a single number measuring both the accuracy and precision of the pose estimates. To illustrate how both accuracy and precision affect the CEP, consider the following examples. Figure 4.7 shows a case where poor accuracy creates the majority of the error. The cross in the upper-right portion of the figure shows the mean error lies near the CEP. If the variance of the estimates is decreased, the mean error would still lie near the circle and the CEP would not change significantly. Figure 4.8 shows a case where poor precision creates the majority of the error. In this case, the mean error is near the origin. Decreasing the variance of the entire population would significantly decrease the CEP.

The ability of the CEP statistic to represent the combined effect of accuracy and precision also obfuscates the role of each source of error. Other error statistics, such as the CEP about the median error can measure precision independently from accuracy. The median error in each dimension can measure accuracy and precision independently in each

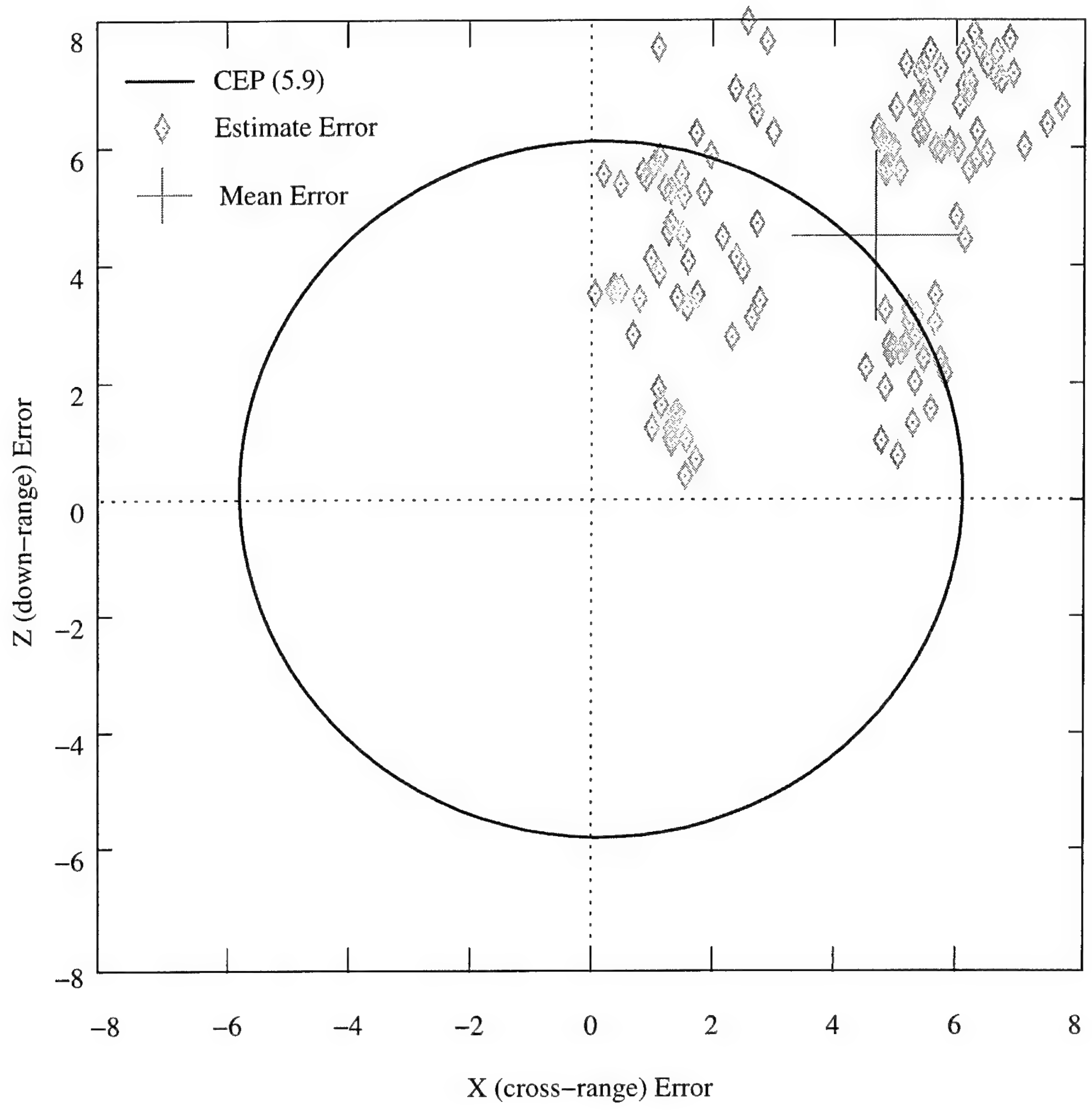


Figure 4.7: CEP due primarily to poor accuracy. The pose estimates are shown as diamonds. The cross represents the mean pose estimate error.

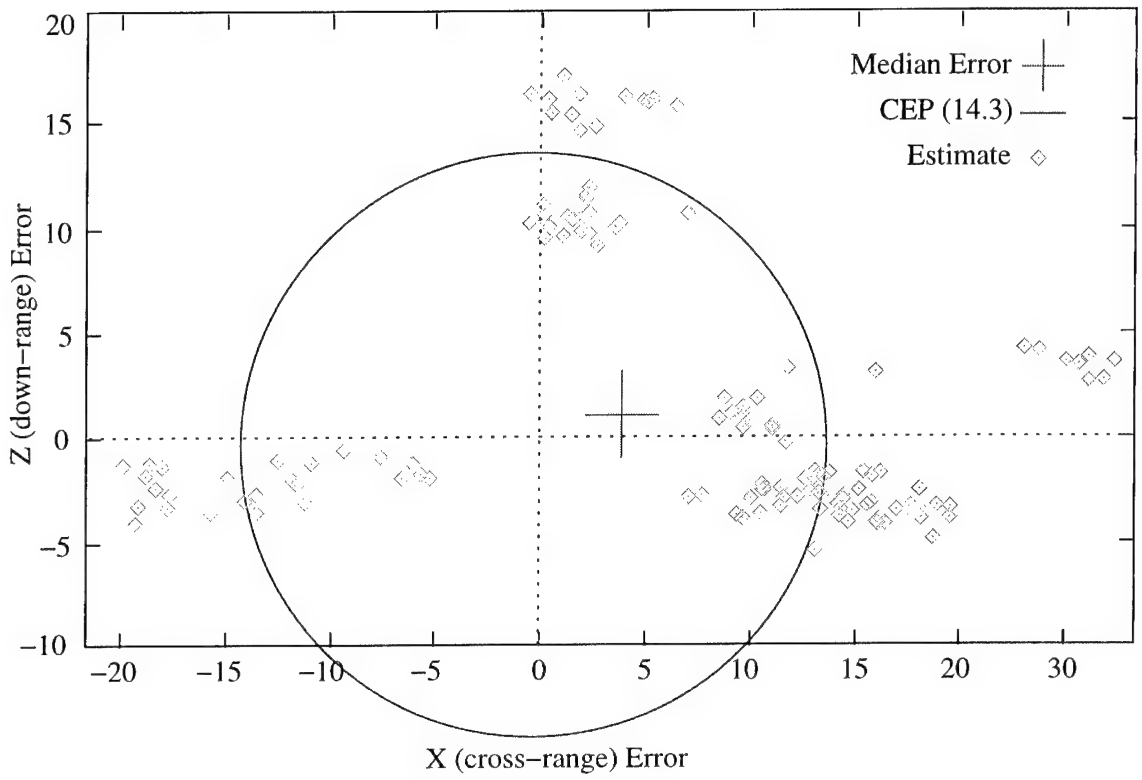


Figure 4.8: CEP due primarily to poor precision. The pose estimates are shown as diamonds. The median pose error is shown as a cross.

dimension. However, using multiple statistics creates an interpretation problem when they disagree. Rather than create a comparison problem, I have elected to provide a single simple statistic measuring both accuracy and precision in all dimensions that requires no distribution assumption. Intuitively, the CEP simply represents a circle, sphere, or hypersphere which contains half of the estimates. The distribution of points is ignored with the assumption that moving the population of estimates closer to the estimate is always desirable.

4.2.2.2 Scaled Circular Error Probable

When a system provides estimates where the precision in one dimension is grossly different than the precision in another dimension, simple Euclidean distance can be misleading. For example, radars frequently provide less precise down-range estimates than cross-range estimates. As a result, the CEP for their estimates is dictated almost exclusively by down-range error and the basic CEP statistic fails to give credit to radars with lower cross-range error.

The basic CEP statistic has a second problem. Estimates with dissimilar dimensions, such as rotation and translation cannot be computed using simple Euclidean distance. To combine rotation and translation dimensions of the estimate, the arbitrary units of measure, such as degrees, radians, or gradians, should not determine the importance of either dimension. If, for example, degrees are used to measure rotation, one degree of rotation would increase CEP in the same manner as one unit of translation. However, if radians are used, approximately 57 degrees (one radian) of rotation would be required to increase CEP in the same manner as one unit of translation. In addition, synthetic environments can be used to construct the same tracking problem at any scale, making the relationship between rotation and translation measurements arbitrary.

Euclidean distance for dissimilar dimensions can be computed using the variance of the measurement in each dimension — commonly called the scaled Euclidean distance. A variation of this technique, which scales and rotates the measurement space, is called the Mahalanobis distance [DH73]. For this application, the three-dimensional scaled Euclidean distance is

$$\sqrt{\frac{(S_x^{True} - S_x)^2}{\sigma_x^2} + \frac{(S_z^{True} - S_z)^2}{\sigma_z^2} + \frac{(S_\theta^{True} - S_\theta^{True})^2}{\sigma_\theta^2}} \tag{4.2}$$

where the values σ_x^2, σ_z^2, and σ_θ^2 are variances in each dimension of the estimate error. The variance in each dimension is computed using all the pose estimates in all the experiments. Using a single set of variances derived from all the problems allows the results from different sub-sets of problems to be compared. When the CEP is computed using the scaled Euclidean distance in place of the Euclidean distance, I call the resulting statistic the scaled CEP.

The scaled CEP, as compared to the CEP, introduces variance as the common unit of measure. In other words, distance is no longer measured in inches or degrees, like the original CEP statistic, but dimension-specific units of variance. Measuring error in units of variance addresses the problems of combining dimensions with grossly different levels of precision and combining dimensions with dissimilar units of measure. For example, given the two radars in the earlier example, consistently imprecise down-range estimates will create a large scaling factor and consistently precise cross-range estimates will create a small scaling factor. As a result, the scaled CEP is no longer dictated exclusively by down-range error and improved cross-range performance will be measurable.

To claim improved accuracy and precision, the scaled CEP using a combination of experts should be less than the scaled CEP using each of the contributing experts individually. Therefore, using the scaled CEP statistic, a relevant hypothesis for the research question is

Hypothesis The scaled CEP for the Extended Condensation Filter's combined pose estimate is less than the scaled CEP using each contributing expert independently.

The render-match algorithm provides an example of the data and analysis required to test this hypothesis. To compute the scaled CEP for the Extended Condensation Filter, using only the render-match expert, the filter is run using only the similarity score w_{RM} and the corresponding probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|w_{RM})$ provided by the render-match algorithm. After the last frame of a movie, the pose estimates for all the particles in the filter are used to compute the scaled CEP. When the training problems are used, the scaled CEP is computed for each of the 150 trained tracking problem — 50 movies with three targets each. The resulting 150 scaled CEP values are used to compute the mean scaled CEP value for the entire problem set. This process is repeated for each of the remaining experts individually. Then the process is repeated for all experts combined. The hypothesis is supported if the mean scaled CEP for all experts combined is less than the mean scaled CEP for each individual expert.

4.2.3 Research Question: Generalization

Research Question Does the Extended Condensation Filter's learning algorithm generalize sufficiently to support operation in new portions of the problem domain?

The issue of generalization is commonly raised for systems that learn responses based on a set of training examples. Can the system perform as well on new (untrained) instances from the problem environment as it did on instances used during training — can the system generalize rather than memorize? As described earlier, only 150 of the 180 tracking problems in problem environment are used to train the filter. Assuming these problems are representative of the problem environment, the look-up tables the filter learned during training should be representative of the probabilities associated with a randomly generated set of movies not seen during training.

Hypothesis When tested on untrained tracking problems, the scaled CEP for the Extended Condensation Filter's combined pose estimate is less than the scaled CEP using each contributing expert independently.

The same scaled CEP statistic used to evaluate the accuracy and precision of the filter when tested on the training problems is used to test the accuracy and precision of the filter when tested on the untrained problems.

Because discrimination requires independent and identically distributed samples, the samples must be collected without running the Extended Condensation Filter — running the filter ensures samples are not independent and identically distributed. Building an accurate ROC curve requires a large number of estimates that hit the ground truth pose $\vec{S} = \vec{S}^{True}$ and a large number of estimates that miss the ground truth pose $\vec{S} \neq \vec{S}^{True}$. As described in Chapter 4, this data is required to estimate the probability of a false positive and true positive given the similarity metric. The training algorithm is forced to provide a large number of each. However, when the filter operates, it quickly forms dense clusters of estimates. These clusters may be near the ground truth pose or far away from it. Therefore, when the filter operates, it may not provide the number of hit or miss estimates required to build an accurate ROC curve. For this reason, discrimination must be tested using the training algorithm, even though the results are not used to train the filter.

4.2.4 Research Question: Is Likelihood Sufficient?

Research Question Is the weight for each particle required to be a probability estimate or, as Pearl indicates [Pea88], is relative likelihood sufficient?

If likelihood is sufficient, the Extended Condensation Filter should still produce a result better than each of the individual contributing experts when the mean sum rule is used to combine expert estimates in place of the product rule. Using the mean sum rule, the discrimination, accuracy and precision using a combination of experts should still be better than the accuracy, discrimination, and precision using each expert independently.

The conditioning, relevance, and causation provided by the formalism of the product rule can be useful for an object recognition system, but likelihood relationships may be sufficient for the Extended Condensation Filter. In other words, the simple statement, "pose $\vec{S}^{True}$ is more likely to be at the pose $\vec{S}$ than any other pose", may be sufficient. This is true for search algorithms and, as discussed in Chapter 3, the Extended Condensation Filter is a search algorithm. If likelihood is sufficient, a large number of algorithms, in addition to the Bayesian product and mean sum rule, might support operation of the filter.

Hypothesis Using the mean sum rule, the AUC of the combined probability estimate is less than the AUC of the probability estimate using each contributing expert independently.

Hypothesis Using the mean sum rule, the scaled CEP for the Extended Condensation Filter combined pose estimate is less than the scaled CEP using each contributing expert independently.

The data collection and analysis using the mean sum rule is identical to the evaluation of the discrimination, accuracy, and precision using the product rule, except the mean sum rule estimate of likelihood $L(\vec{S}|\Delta_0, \ldots, \Delta_N)$ is used to compute the AUC.

4.2.5 Research Question: Convergence

Research Question What are the convergence properties of the filter population? Specifically, how common is a multi-mode distribution of filter particles?

I have elected to test filter populations for the presence of a single cluster of particles for two reasons. First, the assumption is commonly used when extracting an estimate from the filter. Second, if a single mode is commonly present in the filter, the extraction of an estimate is greatly simplified.

If a single cluster of particles is present in the filter, a clustering algorithm should be able to reduce the population to a single cluster. However, a clustering algorithm requires a threshold to measure the similarity of two groups of points. The mean-deviation ratio test is a well-established statistic designed for this purpose [Mur85].

$$\frac{\|\mu_1 - \mu_2\|}{\sigma_1 + \sigma_2} \tag{4.3}$$

where μ_1 and μ_2 are the mean values for two clusters and σ_1 and σ_2 are the standard deviations for two clusters.

Hypothesis The entire population of the Extended Condensation Filter is reduceable to a single statistically similar cluster using a bottom-up hierarchical clustering algorithm.

I use the bottom-up hierarchical clustering algorithm and the mean-deviation ratio test described above to count the number of clusters in the filter [DH73]. Once no clusters should be combined, based on the mean-deviation ratio test, hierarchical clustering stops and the number of clusters are counted. If the count is greater than one, statistically different clusters are present within the filter population. This test will be discussed again when the test results are presented in Chapter 5.

4.3 Extended Condensation Filter Implementation

An implementation of the theoretical models presented in Chapter 3 is required to support empirical testing. Unfortunately, implementations are inherently myopic. Models, in theory, may accommodate an unlimited set of algorithms and parameter settings, but an implementation must select a finite set. In the case of the Extended Condensation Filter, only a finite set of recognition, classification, and feature extraction algorithms can be selected to provide expert estimates — by necessity, a far larger set of algorithms must be ignored. Each expert must also be operated using a single set of parameters. For example, an instance of the Canny edge extractor must operate with a single set of T_{low}, T_{high}, and σ parameters and an instance of a color index algorithm must operate with a single histogram resolution. As a result, implementation of any complex theoretical model only provides information about one tiny portion of a model's potential.

Where possible, the implementation is designed to test the hypotheses, answer the research questions, and accommodate the problem environments. For example, rather than focusing on the absolute performance of the filter, which the next chapter shows is dramatically different depending on the experts selected, the research questions focus on maximizing the benefit of combining multiple sources. As a result, the implementation sets the parameters of the various algorithms to commonly used values, then treats the performance of the recognition, classification, and feature extraction algorithms as a given. This allows me to focus on the improvement of the filter's fused estimate at the expensive of building the best object recognition system.

The selection of the estimate combination operator is critical to my research questions. The product rule and mean sum rule discussed earlier, and alternatives, such as the maximum rule or more sophisticated learning algorithms, such as the neural network or boosting, could each be used to combine the expert's probability estimates. The product, mean sum, and maximum rules are relatively simple and straight-forward to implement. However, as the numerous books and conference proceedings indicate, using a neural network or boosting creates yet another large set of implementation issues. Due to resource constraints and the desire to avoid even further implementation complexity, I have elected to implement two common and simple combination methods — the Bayesian product rule and mean sum rule.

4.3.1 Experts: Contributing Sources

Figure 4.9 illustrates the Extended Condensation Filter implementation used in the remainder of this dissertation. As shown, the implementation uses five separate sources of object recognition information to weight each filter particle — one recognition algorithm, two feature extraction algorithms, and two classifiers. These five experts represent the three basic structural types defined in Chapter 2. In addition, high-level and low-level, object-centered and viewer-centered, global and local, radiometric and geometric methods are represented. The next five sections describe each of these experts in detail.

4.3.1.1 Recognition - Hausdorff-Huttenlocher

As shown at the top of Figure 4.9, the first source of object recognition information is the Hausdorff-Huttenlocher geometric object recognition algorithm [HR92]. The Hausdorff-

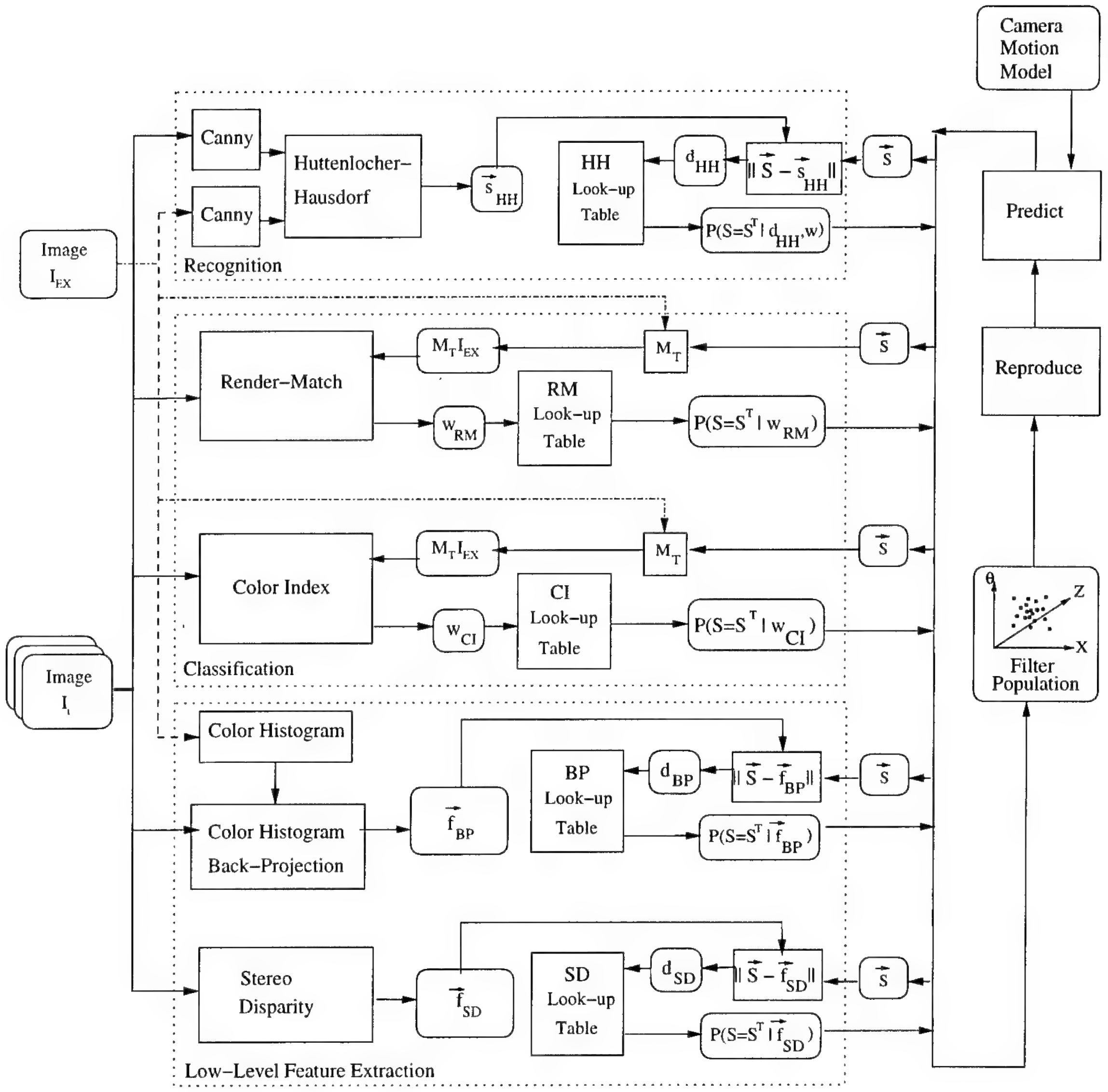


Figure 4.9: Implementation of the Extended Condensation Filter

Huttenlocher algorithm performs object recognition independently of the Extended Condensation Filter. As a result, it provides its own independent pose estimate $\vec{s}_{HH}$. The Hausdorff-Huttenlocher algorithm uses the Hausdorff similarity measure to perform a pose space search for instances of the target object. The algorithm also uses Hausdorff specific heuristics to skip portions of the pose space.

I use the Hausdorff-Huttenlocher algorithm for my implementation for three reasons. First, it provides a contrast with the other algorithms because it relies purely on the geometric features of the problem environment. Second, it is fast. For my implementation, it only requires a few seconds to compute a pose estimate. Third, the algorithm performs object recognition, while the other algorithms perform only classification or feature extraction.

The edge features used by the Hausdorff-Huttenlocher algorithm are extracted from the target image and unknown images using the Canny edge detector with settings $\sigma = 3.0$, $T_{Low} = 0.6$, $T_{High} = 0.9$ [Can86]. The target and input edge features are compared over the user specified range of scale and translation parameters to find the pose that maximizes the number of target and unknown image edge features within the user specified Hausdorff distance. Figure 4.10 illustrates how a range of scale and translation parameters correspond to a camera-relative wedge of pose space starting 1.6 units in front of the camera and extending 10 units from the camera. Searching only in scale and translation space makes this object recognition algorithm very sensitive to out-of-plane rotation.

As described earlier, the Hausdorff distance is defined for each target image I_{EX} edge feature as the distance to the nearest unknown image I_t edge feature. For my implementation the computationally simple city-block distance is used to measure the distance between edge features. If the percentage of matched edge features is above the user specified minimum threshold, the translation and scale values with the maximum percentage of matching edge features are transformed to pose space to compute the target pose $\vec{S}_{HH}$. For this implementation I used a minimum threshold of 90% matching features.

As implemented, the Hausdorff-Huttenlocher algorithm does not consider rotation of the object θ on the ground-plane. Therefore, the minimum Hausdorff distance search takes place in $(u, v, scale)$ space and the result is transformed into the (x, z) pose space. Ground rotation θ is assumed to be zero for all Hausdorff-Huttenlocher estimates. In other words, every Hausdorff-Huttenlocher estimate is assumed to be facing directly toward the camera.

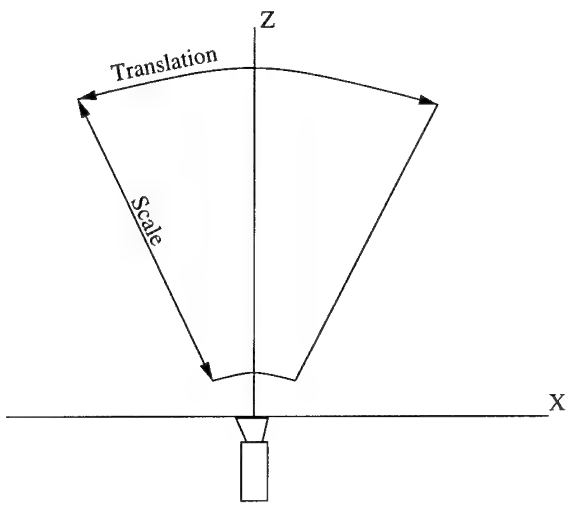


Figure 4.10: The Hausdorff scale and translation parameters correspond to a wedge of camera-relative pose space. The range of scale parameters define the wedge depth. The range of translation parameters define the wedge angle.

The pose is compared to each particle pose in the filter and used to look-up the probability $P(\vec{S} = \vec{S}^{True} | ||\vec{S} - \vec{s}_{HH}||)$. The probability is used to weight each Extended Condensation Filter particle. If no hypothesis with at least 90% matching edge features is identified for an unknown image I_t, the Hausdorff-Huttenlocher algorithm has no effect on the filter update cycle at time t.

Exhaustively searching for the translation and scale parameters that provide the maximum percentage of matching edge features is very time consuming. Huttenlocher's modifications dramatically accelerate the search by estimating the effect of a portion of the translation-scale space on the match score [HR92]. For example, if 20% of the edge features for a given pose have a Hausdorff distance 10 pixels above the threshold, and 90% matching edge features are required to accept a pose, then any translation of the image less than 10 pixels will fail to generate a target estimate. The search and associated measurement of the Hausdorff distance for pose space translations where $\sqrt{(x_{new} - x)^2 + (y_{new} - y)^2} < 10$ can be skipped, where x and y are translation parameters and scale remains constant. Additional short-cuts are used to skip portions of the pose space and accelerate the search.

The Hausdorff-Huttenlocher algorithm performs recognition, so it provides a pose estimate $\vec{s}_{HH}$ and similarity measure w_{HH}. For my implementation, only pose estimates where $w_{HH} \geq 0.9$ (90% matching edge features) are used. The distance d_{HH} between the particle pose $\vec{S}$ and the Hausdorff-Huttenlocher estimate $\vec{s}_{HH}$ has two components, the distance in translation space $\sqrt{(S_x - s_{HHx})^2 + (S_z - s_{HHz})^2}$ and the distance in rotation space $\|S_\theta - s_{HH\theta}\|$. Both components of the distance d_{HH} are used to lookup the probability $P(\vec{S} = \vec{S}^{True}|d_{HH})$ in a two-dimensional histogram, where the translation dimensions is divided into five discrete bins and the rotation dimension is divided into 20 discrete bins. The probability $P(\vec{S} = \vec{S}^{True}|d_{HH})$ is used to weight the particle during the filter population reproduction step.

4.3.1.2 Classification - Render-Match

Both the render-match and color index algorithms shown in the middle of Figure 4.9 are classification algorithms that have been incorporated into the Extended Condensation Filter using the render-match technique [SB01]. They both estimate the appearance of the target object by performing an affine transformation of the target image based on the particle pose $\vec{S}$. In the Figure 4.9, the affine transformation matrix is labeled M_T and the transformed target image is labeled $M_T I_{EX}$. Using either algorithm, the similarity of the relevant portions of the transformed target image and the unknown image are compared. However, render-match and color index use different similarity metrics.

The first classification algorithm, called render-match, compares the images using statistical pixel correlation — the Pearson correlation coefficient. I choose this similarity metric for two reasons. First, pixel correlation is a common and simple way to compare two images. Second, pixel correlation provides a contrast with the second classification algorithm, color index, because it relies on both geometric and radiometric image properties, while color index relies purely on radiometric properties.

As described in the previous chapter, the render-match algorithm measures the similarity between the transformed target image and the unknown image. As shown in Figure 4.11, the target image I_{EX} is mapped onto a polygon and the polygon is transformed based on the particle pose estimate $\vec{S}$. For my implementation, OpenGL is given the estimated target object pose $\vec{S}$ and OpenGL renders the polygon containing the target image at that pose.[5] Then pixel-level correlation is performed to compare the relevant RGB values in the transformed target image MI_{EX} and the unknown image I_t and produce the render-match similarity measure.

[5]Brian Paul's Mesa OpenGL 3.0 is used for my implementation.

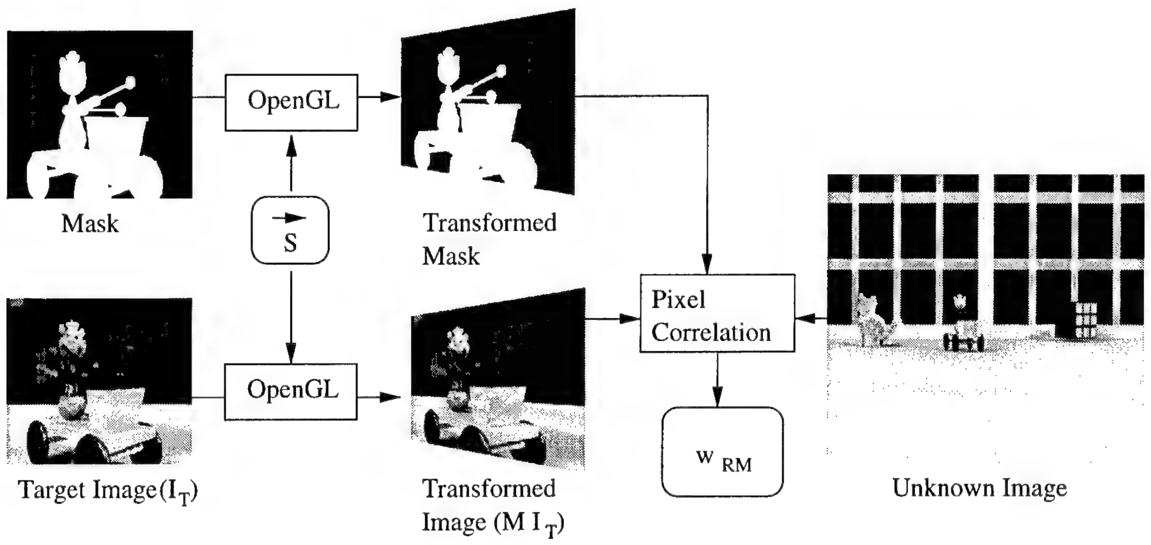


Figure 4.11: Implementation of the render-match algorithm using OpenGL. Note, the same process is used to implement the color index algorithm, except "Pixel Correlation" is replaced with "Color Index."

To accommodate target objects that do not fit neatly on a rectangular polygon, a binary image identifying the relevant target pixels is used as a mask, as shown in Figure 4.11. The target image mask pixels are set to true, if the corresponding pixel in the target image should be included in the correlation computation. As shown in Figure 4.11, the binary mask is transformed using OpenGL and the same particle pose $\vec{S}$ used to transform the target image. Only the pixels in the unknown image and transformed target image that correspond to the white (true) pixels in the mask are compared.

Despite the use of OpenGL, rendering the target image at each point in pose space is time consuming in my implementation. The mean run times shown later in Figure 5.19 on page 184 show the classifiers consume over 90% of the wall-clock time per filter iteration. However, rendering the target image is only performed once regardless of the number of classification algorithms used in an implementation. Also, commodity video hardware is available to dramatically accelerate OpenGL library calls, but is not used for my implementation.

The result of the render-match classification algorithm is the correlation value w_{RM} for each particle pose $\vec{S}$. The correlation value is used to look-up the probability $P(\vec{S} = \vec{S}^{True}|w_{RM})$ in a one-dimensional histogram composed of 20 discrete bins. The probability is used to weight the filter particles for reproduction.

4.3.1.3 Classification - Color Index

The color index similarity metric was originally designed by Swain to compare the color content of two images and support visual database queries [SB91]. Intuitively, color index answers the question, "Are the unknown image's colors similar to the target image's colors?" My application of the Color Index algorithm compares the color content of the transformed target MI_{EX} and unknown images using a process similar to that shown in Figure 4.11 — the only difference is the similarity metric. In fact, to speed-up the weighting of each particle, the transformed target and mask images created for the render-match classifier are used for the color index classifier.

I use color index as a classifier for two reasons. First, color index provides a second classifier that can be compared to the pixel-correlation used by render-match. Second, color index provides a contrast to the render-match classifier because it is entirely dependent on the color content of the problem environment.

Using the transformed target image and mask created for the render-match classifier, the color index is computed for the portions of the target and unknown images identified by the transformed target mask

$$ColorIndex = \frac{\sum_{R=(0,16)} \sum_{G=(0,16)} \sum_{B=(0,16)} min(H_T([R][G][B]), H_R([R][G][B]))}{PixelCount_R} \quad (4.4)$$

where H_R is the RGB histogram for the applicable region of the unknown image, H_T is the RGB histogram for the applicable region of the transformed target, and $PixelCount_R$ is the total number of true pixels in the target image mask after transformation. Each of the histograms is a three-dimensional 16x16x16 discrete histogram of RGB values. Note, the two histograms are created dynamically using just the pixels indicated by the binary mask. Therefore, both histograms have the same number of total entries. Using my implementation's 16x16x16 histogram quantization (4096 bins), a pixel with the RGB values 123, 200, 75 adds one count to the histogram bin [7][12][4] — 123 mod 16 is 7, 200 mod 16 is 12, and 75 mod 16 is 4.

The result of the color index classification algorithm is the normalized color index score w_{CI}, where 1.0 is a perfect match in color space. The color index is used to look-up the probability $P(\vec{S} = \vec{S}^{True} | w_{CI})$ in a one-dimensional histogram composed of 20 discrete bins. The probability is used to weight the filter particles for reproduction.

4.3.1.4 Low-Level Feature - Stereo Disparity

Stereo disparity does not perform object recognition or classification — it merely identifies range features. In fact, Figure 4.9 shows the stereo disparity algorithm knows nothing about the target object. I use stereo disparity in my implementation for two reasons. First, it does not require a new sensor model, such as radar or sonar would, to simulate range feature extraction. Stereo disparity only requires a second camera model. Second, range features provide an entirely different type of feature than any of the other algorithms.

One of the recurring problems for stereo disparity is quickly and accurately identifying good candidate features for measuring the disparity between two images — what Duda and Hart call the correspondence problem [DH73, TV98]. My implementation uses a three step process to identify candidate features and measure stereo disparity.

First, a 3x3 horizontal derivative kernel is run over the left and right images. Pixel groups with a total derivative less than the threshold are eliminated from further consideration. This step is good at eliminating large uniform areas of the image that make poor candidate features. Unfortunately, it can also eliminate very small uniform areas that make good candidate features.

Second, each remaining candidate feature in the right image is compared to each remaining candidate feature in the left image, but only within the horopter. The line from the left sensor to the matching feature on the left image plane defines the left epipolar line. The line from the right sensor to the matching feature on the right image plane defines the right epipolar line. The point where these two lines intersect is the epipole — the estimated pose of the feature in three-dimensional space. The horopter defines the maximum disparity, or distance in pixels, between the point where the left epipolar line intersects the left image plane and right epipolar line intersects the right image plane.

The horopter defines a maximum and minimum displacement horizontally and vertically in the left image relative to a pixel in the right image. For example, using a horopter of width 10 and height 4, a feature in the right image centered at $(u = 20, v = 30)$ creates a horopter in the left image with upper left corner at $(u = 10, v = 32)$ and lower right corner at $(u = 20, v = 28)$. Note the horopter is always to the left of the candidate feature's position because the matching feature in the left image must be to the left of the feature in the right image. You can test this assumption by closing your left eye, then lining up a close and distant object. Quickly closing your right eye and opening your left eye causes the distant object to jump to the left relative to the close object.

The size of the horopter limits the maximum and minimum displacement the stereo disparity filter can measure — the left epipolar line cannot lie outside the horopter. As a direct consequence, the horopter also limits the maximum and minimum depth the implementation can measure. The maximum and minimum can be increased, but the resolution and size of the image impose practical limitations. Features that are too far away are not clearly distinguishable and features that are too close have disparity values exceeding the width of the image. Larger horopters also require more search.

You can play with different horopters using the previous eye closing and opening technique. Close your left eye and line up the same close and distant objects. Put your hands up to create a frame for the objects. When you close your right eye and open your left, the distant object may jump outside of the frame. If it does, your "algorithm" uses too small a horopter to measure this disparity. You can increase or decrease the distance between your hands to experiment with different horopters.

The horopter also constrains the location of target objects in the problem environment. If objects are too close or too far, they fall outside the horopter and stereo disparity fails. If an object is present in the right image but jumps outside the left image, stereo disparity fails. The target pose constraints listed in Table 4.1 on page 106 and Table 4.2 on page 110 are designed to minimize this problem.

Finally, pairs of features with correlation above a minimum threshold are used to compute the range estimates, as described in Chapter 2. Only one-to-one correspondences are allowed between features. These features are stored in a two-dimensional (x, z) array. Each particle $\vec{S}$ in the filter is compared to every stereo disparity feature in the array to find the minimum distance to a disparity feature for each particle. The particle's minimum distance is used to look-up the probability $P(\vec{S} = \vec{S}^{True}|d_{SD})$ in a one-dimensional histogram composed of 16 discrete bins. The probability is used to weight the filter particles for reproduction.

4.3.1.5 Low-Level Feature - Color Histogram Back-Projection

My color histogram back-projection implementation, or back-projection for short, is based on Swain's color index [SB91] and suggestions from Bruce Draper. Back-projection is dif-

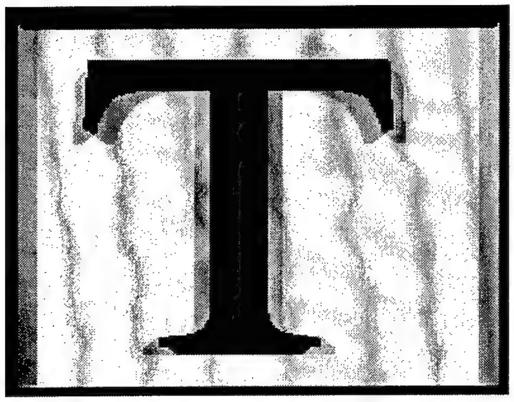


Figure 4.12: Target image I_{EX} used for the Color Histogram Back-Projection example.

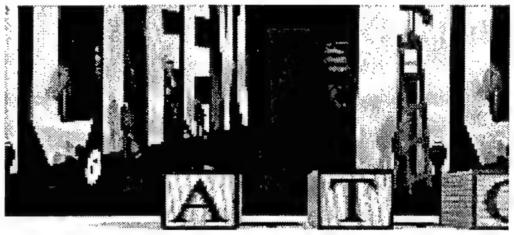


Figure 4.13: Unknown image I_t used for the Color Histogram Back-Projection example.

ferent from the color index algorithm because it compares each unknown image pixel to the entire target image rather than comparing image to image — back-projection classifies pixels, while color index classifies images. Intuitively, color histogram back-projection answers the question, "Is this unknown pixel's RGB value present in the target image?" If the answer is yes, a new back-projection feature vector $\vec{f}_{BP}$ is created at the pixel's location.

I use back-projection in my implementation for two reasons. First, it provides a low-level feature that is very different from stereo disparity. Back-projection features are dependent on the target image, while stereo disparity features are not. Back-projection features are represented on the (u, v) image plane, while stereo disparity features are represented in (x, z) pose space. Second, back-projection runs very quickly, requiring only one or two tenths of a second per filter update for my implementation.

The back-projection algorithm computes a three-dimensional color histogram for the target image and each unknown image. The RGB content of the image forms each of the dimensions of a three-dimensional 16x16x16 histogram. A ratio histogram is computed from a target image histogram and each unknown image histogram using

$$H_R([R][G][B]) = min(1, \frac{H_{EX}([R][G][B])}{H_t([R][G][B])} \tag{4.5}$$

where H_{EX} is a target image histogram that always contains 320x240 entries and H_t is the input image histogram that always contains 320x240 entries — 320x240 is the standard image size throughout the implementation. An example target and unknown image are shown in Figures 4.12 and 4.13. If the ratio is greater than one, the ratio histogram is set

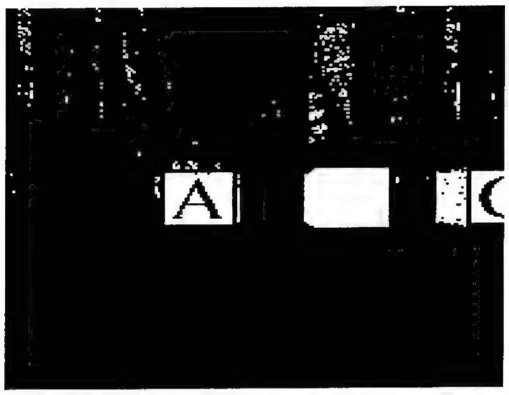


Figure 4.14: Typical features (white pixels) identified by Color Histogram Back-Projection. Note the large number of extraneous features above the blocks.

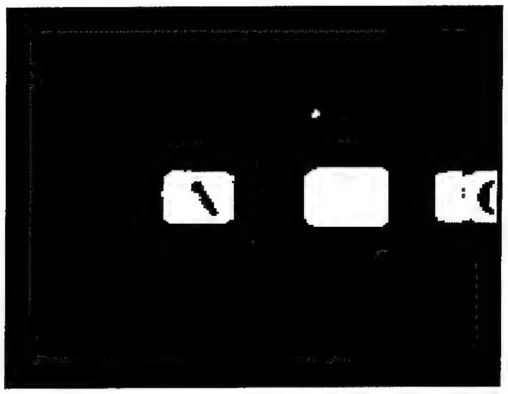


Figure 4.15: Typical Color Histogram Back-Projection results after 5x5 median filter is applied. Note the the region of extraneous features remaining above the middle block.

to one. After computing the ratio histogram, the unknown image pixel values are used as indices into the ratio histogram and every pixel value is replaced with its corresponding ratio histogram value. If the ratio histogram value is one, the pixel is set to true — the unknown image pixel is assumed to be part of the target image. If the value is less than one, the pixel is set to false — not part of the image. The result is a binary image identifying matching pixels as shown in Figure 4.14. A median filter is run over the binary image to eliminate isolated pixels, as shown in Figure 4.15. Each true pixel becomes a back-projection feature vector $\vec{f}_{BP}$. Reducing the entire target image to an array of 4096 integers combined with a single pass through the unknown image makes the back-projection algorithm the least time consuming source of information.

Once the back-projection features are identified, each particle's pose $\vec{S}$ is projected onto the camera view plane and compared to every back-projection feature to find the minimum distance d_{BP} to a back-projection feature. This process is shown in Figure 4.9. The particle's minimum distance is used to look-up the probability $P(\vec{S} = \vec{S}^{True} | d_{BP})$ in a one-dimensional histogram composed of 16 discrete bins. The probability is used to weight the filter particles for reproduction.

4.3.2 Combining Expert Estimates

Once each source of object recognition information, or expert, has provided the similarity measure w_i, the corresponding learned probability data is retrieved from the look-up tables and combined to weight each particle. As shown in Figure 4.9, obtaining each expert's probability estimate is part of the larger Extended Condensation Filter update cycle — the large vertical loop on the right side of the figure.

The probabilities from each expert are used differently depending on the combination rule employed. If the product rule is used, each weight Δ_i, obtained from the five sources, is used to look up the probabilities $\hat{P}(\Delta_i|\vec{S} = \vec{S})$ and $\hat{P}(\Delta_i)$. The probabilities are used to compute the ratio

$$\frac{\hat{P}(\Delta_i|S = S^{True})}{\hat{P}(\Delta_i)} \tag{4.6}$$

and the combined probability is multiplied by each expert's ratio in turn

$$\hat{P}(S = S^{True}|\Delta_0, \ldots, \Delta_N) = \hat{P}(S = S^{True}) \prod_{i=1}^{N} \frac{\hat{P}(\Delta_i|S = S^{True})}{\hat{P}(\Delta_i)} \tag{4.7}$$

where each of the $(\Delta_0, \ldots, \Delta_N)$ estimates correspond to the color histogram back-projection, color index, Hausdorff-Huttenlocher, render-match, and stereo disparity algorithm.

If the mean sum rule is used, these same probabilities are used to compute the probability estimate

$$P(S = S^{True}|\Delta_i) = \frac{P(S = S^{True})P(\Delta_i|S = S^{True})}{P(\Delta_i)} \tag{4.8}$$

then the individual estimates are averaged

$$L(S = S^{True}|\Delta_0, \ldots, \Delta_N) = \frac{\sum_{i=0}^{N} P(S = S^{True}|\Delta_i)}{N} \tag{4.9}$$

It is possible for the Extended Condensation Filter to encounter conditions different from any condition observed during training. In these conditions, it is possible for an expert to generate a similarity measure Δ_i that was not observed during training. If this happens, the probabilities associated with the new measures are unknown and the look-up table value for $P(\Delta_i)$ is zero. In this case, the ratio

$$\frac{\hat{P}(\Delta_i|S = S^{True})}{\hat{P}(\Delta_i)} \tag{4.10}$$

is undefined. In this case, the expert is eliminated from the combined product or mean sum rule estimate because the expert's lack of information prevents an informed estimate.

4.3.3 Particle Reproduction

Once each of the 200 particles contained in the Extended Condensation Filter implementation is assigned a single combined estimate of the probability $\hat{P}(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N)$ or $L(\vec{S} = \vec{S}^{True}|\Delta_0, \ldots, \Delta_N)$, weighted random reproduction is performed.

During the particle weighting step, the total weight of all the particles in the population is computed. This total weight is used to generate a uniform random number between 0 and the total weight. This random number is used to perform a binary search in the population for a specific member of the filter population. This is similar to a lottery selection where a population member with a larger number of tickets has a greater chance of selection. In the case of my implementation, particles with larger weights occupy more of the total range of random numbers, so they have a greater chance of selection. Binary search is an efficient match with this data structure, finding the randomly selected population particle in the typical $O(\log_2(N))$ time associated with binary search.

4.3.4 Prediction

The next step in the filter update cycle, prediction, moves the particles according to the motion model. In all of the problem environments the camera motion is known. One might mistakenly assume adding uncertainty to particle positions is not required. However, motion uncertainty acts like mutation in an evolution strategy. If no uncertainty is added to the model, the population converges to a single minima, as defined by the mean-deviation ratio earlier, and never explores other portions of the pose space.

For my implementation, I apply the following camera motion model to predict the next location of each particle

$$\begin{bmatrix} x_{i+1} \\ z_{i+1} \\ \rho_{i+1} \end{bmatrix} = \begin{bmatrix} x_i \\ z_i \\ \rho_i \end{bmatrix} + \begin{bmatrix} Gaussian(0,1) \\ v_z + Gaussian(0,1) \\ Gaussian(0,1) \end{bmatrix} \tag{4.11}$$

where ρ is the camera ground-plane rotation about the y-axis, or "yaw" in aircraft terminology. This motion model simply represents the camera moving forward into the scene along the z-axis with no rotation.

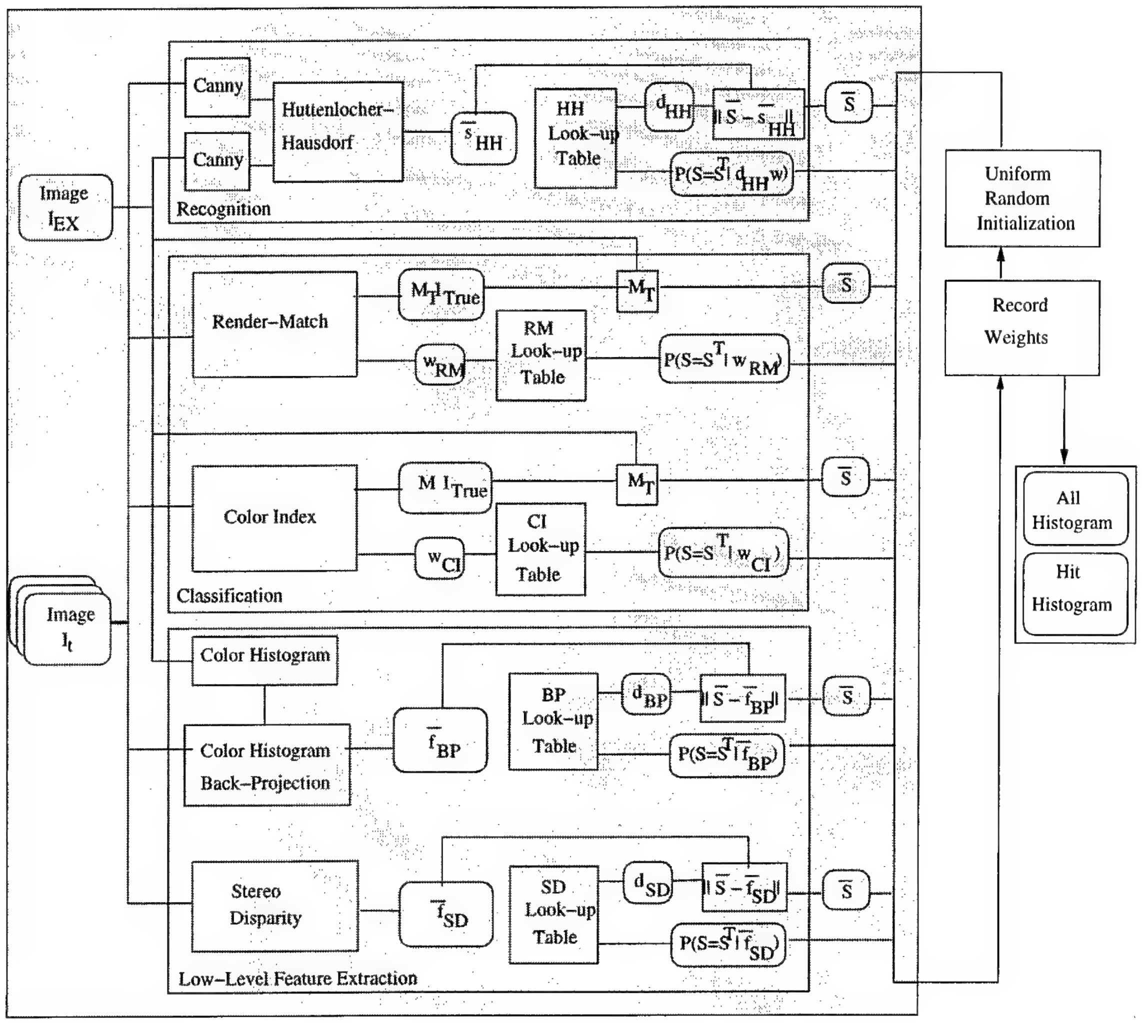


Figure 4.16: Implementation of the learning algorithm. The portion of the implementation shown in gray is identical to the filter.

4.3.5 Learning the Probability Estimates

Implementation of the software required to build the look-up tables is very similar to the Extended Condensation Filter implementation. Figure 4.16 shows only two changes are required to turn my implementation of the Extended Condensation Filter into an implementation of the learning algorithm. The filter's prediction step is replaced with a uniform random initialization step and the reproduction step is replaced with the collection of the particle weights into two histograms. The simplicity of these modifications is a real advantage, in terms of implementation, for the Extended Condensation Filter.

As described in Chapter 3, two training phases are required to efficiently collect the data to build accurate probability look-up tables. The fundamental steps shown in Figure 4.16 are the same for both phases of the learning implementation, only the random initialization phase changes. Either the entire pose space is sampled or the pose space near the ground truth where $\vec{S} = \vec{S}^{True}$ is sampled. As discussed in Chapter 3, the two phase sampling process dramatically improves the probability estimate while reducing the number of samples required. In either phase, the steps are uniform random initialization, particle weighting, hit or miss evaluation (only required when sampling the entire pose space), and tallying the instances of the particle weights in the appropriate histogram.

4.4 Implementation Questions

The research questions posed at the end of Chapter 3 address general properties of the Extended Condensation Filter. My implementation of the Extended Condensation Filter suggests additional questions. Unfortunately, only a few of the most interesting questions can be adequately investigated due to resource constraints. This section describes two implementation specific questions and a corresponding set of empirically testable hypotheses.

4.4.1 Implementation Question: Two-Dimensional Target Model

Implementation Question Does the Extended Condensation Filter implementation degrade gracefully in the presence of objects with geometry poorly modeled by a single polygon?

To accommodate classification algorithms, the general Extended Condensation Filter requires the projection of a target model into pose space. My implementation of this process using a single polygon is illustrated in Figure 4.11. However, any model compatible with projection into pose space could be used. For example, a high-resolution three-dimensional geometric model accompanied by multi-spectral surface textures would provide a compatible and powerful model for the Extended Condensation Filter.

Due to limited resources, I use a single polygon to represent my target objects. My model is composed of a target image mapped onto a single polygon, a mask image mapped onto a single polygon, and the associated camera parameters. This simple model creates increasing error as a function of out-of-plane rotation, especially for objects with geometry that is not well represented by a single polygon.

The nearly two-dimensional target views used in Block World One and Block World Two are ideally suited to a model using only a single polygon. However, the pig, Rubik's cube, and cat targets used in Toy World are poorly suited to a single polygon model. The pig's curved geometry and self-occlusion create modeling error. The edge-on view of the Rubik's cube creates modeling error due to the two cube faces receding into the image. The cat's curved geometry and self-occlusion create modeling error.

Modeling error created by the application of the single polygon model in Toy World should not cause the Extended Condensation Filter to fail. The use of other experts that do not rely on the polygonal model and the process of learning the inaccuracies associated with the modeling process should allow the filter to degrade gracefully. In other words, the filter's combined estimate should still be more discriminating, accurate, and precise than any of the contributing expert estimates despite the errors created by the single polygon target model.

Hypothesis Within Toy World, the AUC for the Extended Condensation Filter's combined probability estimate is less than the AUC for the probability estimate using each contributing expert independently.

Hypothesis Within Toy World, the scaled CEP for the Extended Condensation Filter combined pose estimate is less than the scaled CEP using each contributing expert independently.

Using probability and pose estimate data from Toy World, the discrimination, accuracy, and precision are evaluated using the same AUC and scaled CEP statistics described earlier.

4.4.2 Implementation Question: Importance of Color

Implementation Question Does color provide the best source of discrimination in the synthetic problem environments?

Block World One, Block World Two, and Toy World are very colorful. Given these problem environments, color may provide the best way to discriminate between the target object and non-target objects. The algorithms which are highly dependent on color, such as color index and color histogram back-projection, should have a clear advantage, if color provides the best discrimination. Algorithms which take less advantage of color, such as Hausdorff-Huttenlocher and stereo-disparity should be at a disadvantage. If color provides the best form of discrimination, the evaluation statistics should indicate better performance by the algorithms that rely on color.

Hypothesis The AUC for the color histogram and back-projection algorithms is less than the AUC for the Hausdorff-Huttenlocher and stereo disparity algorithms.

Hypothesis The scaled CEP for the color histogram and back-projection algorithms is less than the scaled CEP for the Hausdorff-Huttenlocher and stereo disparity algorithms.

The data required to test these hypotheses is identical to the previous evaluations of discrimination, accuracy, and precision. However, the analysis will compare the AUC and scaled CEP statistics of individual experts.

4.5 Summary

In the process of building a bridge between theory and empirical testing, this chapter has assembled a set of tools. Block World One, Block World Two, and Toy World provide easily controlled and examined test environments, the hypotheses provide focused statements and data collection goals designed to support answers to the research questions, and the Extended Condensation Filter implementation provides a functional test article for experimentation. The next chapter shows the results of employing the filter implementation in the problem environments with the goal of testing the hypotheses and, as a result, shedding light on the research and implementation questions.

Chapter 5

Results

This chapter puts the Extended Condensation Filter to the test. Specifically, the implementation described in Chapter 4 is tested using 540 random problems from the three problem environments described in Chapter 4. The analysis of the test results indicates, with only a few limited and predictable exceptions, that the Extended Condensation Filter, using the Bayesian product rule, provides greater accuracy and precision than any of the contributing experts. The results also indicate the mean sum rule generally provides better discrimination than the Bayesian product rule, while it fails to consistently provide better accuracy and precision.

This chapter is organized around the research questions and hypotheses in the previous chapter. However, some of the research questions and hypotheses use the same data and require similar analysis. To reduce repetition, I have grouped the results and analysis into six sections: product rule discrimination, product rule accuracy on the training problems, mean sum rule accuracy on the untrained problems, mean sum rule discrimination, mean sum rule accuracy on the untrained problems, and convergence.

Table 5.1 makes the relationship between the sections in the previous chapter and this chapter explicit. Each row represents a section from this chapter and each column represent a section from Chapter 4. From left to right, the columns are: discrimination and accuracy and precision, generalization from training, the sufficiency of likelihood, convergence, two-dimensional model, and sufficiency of color. In the rows, "trained" indicates the 450 training problems are used, "untrained" indicates the 90 problems withheld from training are used, Π indicates the Bayesian product rule is used, Σ indicates the mean sum rule is used, and "Cluster Test" indicates the bottom-up hierarchical clustering test is used. An "X"

Table 5.1: Relationship between the results in this chapter (rows) and the questions in Chapter 4 (columns). The column and row headings are described in the text.

			Chapter 4: Questions						
			Discrim.	Accuracy	Gen.	Like.	Converg.	2D	Color
		Section	4.2.1	4.2.2	4.2.3	4.2.4	4.2.5	4.4.1	4.4.2
Chapter 5: Results	Discrim.,	5.1.1	X		X			X	X
	untrained,	5.1.2	X		X				X
	Π rule	5.1.4	X		X				
	Accuracy,	5.2.1		X				X	X
	trained,	5.2.2		X					
	Π rule							X	X
	Accuracy,	5.2.3		X	X			X	X
	untrained,								
	Π rule								
	Discrim.,	5.3.1	X		X	X		X	X
	untrained,				X				
	Σ rule				X				
	Accuracy,	5.3.2		X	X	X		X	X
	untrained,								
	Σ rule								
	Cluster	5.4.1					X		
	Test	5.4.3					X		

indicates the data in the section from this chapter is relevant to the corresponding section in Chapter 4.

The 540 tracking problems containing 17,820 distinct movie frames are used to generate 33,264,000 pose estimates, excluding pose estimates generated during training. This large set of data can be partitioned in many ways and each partitioning provides different insight to the filter's performance. For example, probability estimates for all the frames of each tracking problem can be used to create 540 ROC curves and 540 corresponding AUC estimates. This tracking problem partitioning tests discrimination in response to different target objects in different environments. Results reported later show the filter responds very differently to different problem environments and targets. Alternatively, probability estimates from each of the 33 frames across all 540 tracking problems can be used to create

33 ROC curves and 33 corresponding AUC estimates. This frame partitioning compares discrimination when the camera is at different locations relative to the target objects.

Different data partitions also test slightly different versions of the hypotheses presented in the previous chapter. For example, the following hypothesis was originally posed in Chapter 4:

Hypothesis The AUC of the combined probability estimate is less than the AUC for the probability estimate using each contributing expert independently.

This hypothesis can be tested using different partitionings of the data. For example, using the hypothesis:

Variation One For all the tracking problems in the three problem domains, the AUC of the combined probability estimate is on average less than the AUC using each expert independently.

This variant tests the original hypothesis using all the frames in each of the tracking problems. The original hypothesis can also be tested using all the estimates from a single frame using this hypothesis:

Variation Two For all the camera positions, the AUC of the combined probability estimate is less than the AUC using each expert independently.

These variants of the hypotheses can be illuminating, but an exhaustive report on all possible variants would be just that: exhausting. During my examination of the different data partitions, I used combinations of six different parameters: experts used, training problems versus untrained problems, environment, movie, target, and frame. To provide a general understanding of all the results, while avoiding pages and pages of tables, I provide a single representative partitioning of the data for each topic, then highlight especially illustrative results or any anomalous results discovered in the remaining partitions. Where an anomaly occurs, I present more detailed results to explore possible reasons for the anomaly. For example, in the first section I show improved discrimination using a tracking problem partitioning. Then an anomaly in a different partitioning of the results is identified and explored in detail.

A primary issue in machine learning is the ability of a system to generalize to new instances of a problem, rather than memorize a set of responses [Mit97, SB98]. For this reason, this chapter focuses on the untrained problems. This chapter also focuses on the untrained problems because testing all 450 of the less interesting training problems requires far more computer resources than testing only the 90 untrained problems. For these reasons, the accuracy of only one combination of experts is tested using the training problems while 31 combinations are tested using the untrained problems. In addition, the accuracy of the mean sum rule is only tested using the untrained problems.

5.1 Discrimination

This section looks at the discrimination provided by the Extended Condensation Filter using the Bayesian product rule. Discrimination is not a property of the Extended Condensation Filter, but a property of the information provided by the experts, learning the conditional target pose probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|\Delta)$, and the expert combination rule. Therefore, it is not necessary to run the Extended Condensation Filter to measure the AUC and evaluate discrimination. In fact, normal operation of the filter prevents collection of the data need to compute the AUC.

Discrimination cannot be tested using particle data from normal filter operation. Building an accurate ROC curve requires a large number of estimates that hit the ground truth pose $\vec{S} = \vec{S}^{True}$ and a large number of estimates that miss the ground truth pose $\vec{S} \neq \vec{S}^{True}$. As described in Chapter 4, both the hit and miss data are required to estimate the probability of a false positive and true positive given the similarity metric. To populate the two histograms, a large number of each type of sample must be recorded. However, when the filter operates, the filter quickly forms dense clusters of estimates. These clusters may be near the ground truth pose or far away from it. Therefore, when the filter operates, it may not provide the number of hit or miss estimates required to build an accurate ROC curve. For this reason all the discrimination results are always derived by running the filter in the training mode described in Chapter 3. Although the training algorithm is run on the 90 untrained tracking problems to collect the data needed to test discrimination, the data is not used to build the look-up tables — this prevents testing on the training data.

There are 31 possible combinations of experts tested in this chapter. There are 31 combinations because each expert can either be used or not used — providing a power set of 2^5 possible combinations. However, the combination of no experts is not reported because it provides no discrimination — an AUC of 0.5. Both no experts and a perfect expert were used to verify the data collection and analysis process. Each of the 31 comparisons uses the AUC obtained when the conditional probability estimate $\hat{P}(\vec{S} = \vec{S}^{True} | \Delta_0 \ldots \Delta_N)$ is used to discriminate between target poses $\vec{S} = \vec{S}^{True}$ and non-target poses $\vec{S} \neq \vec{S}^{True}$, where $\Delta_0 \ldots \Delta_N$ represents one combination of information sources from the 31 combinations of experts. Each mean AUC reported in this section is derived by taking the mean value of the 90 untrained tracking problem AUC values.

The results in this section support the following hypothesis originally posed in Section 4.2.1:

Hypothesis The AUC for the combined estimate is less than the AUC for the estimate using each expert independently.

Only one exception is noted. The exception occurs when the independence assumption used to derive the Bayesian product rule is violated.

5.1.1 Discrimination by Tracking Problem

Table 5.2 compares the discrimination of the individual experts to all possible combinations of experts using the 90 untrained problems. The first two rows list each of the five individual experts and the discrimination (AUC) they provide. The first two columns list all 31 possible combinations of experts and the discrimination they provide. The middle of the table tests the statistical significance of the difference between the row and the column.

The first five rows of results, before the double line, compare the discrimination of individual experts. Note, the first five rows of the table are symmetric along the diagonal. As described in Chapter 4, the AUC measures the discrimination on a scale from 0 to 0.5, where 0 is perfect discrimination and 0.5 is no discrimination. Table 5.2 uses a tracking problem partitioning, where each mean AUC value represents the average discrimination across all 90 untrained problems. Each expert is run on each of the 90 untrained problems. As described in Chapter 4, all the particle weights collected in all 33 frames of each tracking problem are used to build a non-target and target distribution. These two distributions are used to compute an AUC value. These 90 AUC values are used to compute the mean AUC shown in the second row and second column of the table. As a result, the mean AUC values in Table 5.2 are a very broad representation of discrimination that includes all the untrained problems in all the problem domains.

The rest of the table compares a combination of two or more experts to each of the contributing experts. To estimate the AUC for a combination of two or more experts, the probability estimate is computed using the Bayesian product rule, then the AUC is computed using the new combined conditional probability estimate. For example, to estimate the probability $P(\vec{S} = \vec{S}^{True})$ conditioned on both back-projection (BP) and color index (CI), the Bayesian product rule yields

$$\hat{P}(\vec{S} = \vec{S}^{True}|w_{BP}, w_{CI}) = \hat{P}(\vec{S} = \vec{S}^{True})\frac{\hat{P}(w_{BP}|\vec{S} = \vec{S}^{True})\hat{P}(w_{CI}|\vec{S} = \vec{S}^{True})}{\hat{P}(w_{BP})\hat{P}(w_{CI})} \tag{5.1}$$

This combined probability estimate is used, just like the individual experts, to build an ROC curve and estimate the AUC. For example, the eighth row in Table 5.2 shows the discrimination using color histogram back-projection and color index combined (BP, CI) is 0.2018. The column for color histogram back-projection (BP) alone shows a discrimination of 0.2872 and color index (CI) alone shows a discrimination of 0.2365. The combined discrimination is better than each contributing expert.

The middle section of the table indicates the statistical significance of the difference between the mean AUC value in the row and the mean AUC value in the column. A * indicates none of the non-parametric paired sample trials, as described in Appendix C, provided a negative mean difference. Since the non-parametric test uses 1,000 trials, a * means the individual confidence level is greater than 99.9% that the two values do not represent a distribution with the same mean value. The paired sample test and Bonferoni adjustment are described in Appendix C. When the column-row intersection is blank, the comparison is irrelevant.

The results in Table 5.2 support several observations. First, only in the case of color index (CI) and Hausdorff-Huttenlocher (HH), where the individual confidence level is 77.9%, is the difference between two combinations statistically ambiguous. Second, the results indicate stereo disparity is far more discriminating than any other expert — a surprising result considering stereo disparity knows nothing about the target object. Hausdorff-Huttenlocher appears to be the least discriminating. However, frame partitioning of the results in the next section shows the Hausdorff-Huttenlocher expert is far more discriminating when the camera moves closer to the target object. Third, every combination of experts appears to be more discriminating than each of the contributing experts.

Figure 5.1 provides a different view of the results shown in Table 5.2. Rather than comparing the performance of each combination of experts to the individual experts, Figure 5.1 highlights the relative performance of various combinations of experts. The figure supports two observations. First, the stereo disparity expert dominates all the combinations of experts. Every combination that includes stereo disparity is always several times better than any combination that does not include stereo disparity. Second, simply adding another expert to a combination does not necessarily improve performance. This point will be discussed again later in this section. Figure 5.1 will be used again later when comparing the discrimination provided by the Bayesian product rule to the discrimination provided by the mean sum rule.

5.1.2 Discrimination by Frame

The previous section examined discrimination using a tracking problem partitioning. This section examines discrimination using a frame partitioning. Rather than compute an AUC using all the estimates associated with a tracking problem, this section computes an AUC using all the estimates for only one of the 33 frames — each AUC is composed of estimates across multiple tracking problems. In general, the results confirm the increased discrimination observed in the previous section and, more importantly, illustrate one way experts interact to improve discrimination.

Table 5.2: Discrimination provided by all the expert combinations. The experts are back-projection (BP), color index (CI), Hausdorff-Huttenlocher (HH), render-match (RM), and stereo disparity (SD).

Expert(s)		BP	CI	HH	RM	SD
	Mean AUC	0.2872	0.2365	0.2911	0.2752	0.06855
BP	0.2872	n/a	*	77.9%	*	*
CI	0.2365		n/a	*	*	*
HH	0.2911			n/a	*	*
RM	0.2752				n/a	*
SD	0.06855					n/a
BP, CI	0.2018	*	99.8%			
BP HH	0.1589	*		*		
BP, RM	0.2208	*			*	
BP, SD	0.05345	*				*
CI, HH	0.1581		*	*		
CI, RM	0.2144		*		*	
CI, SD	0.05511		*			*
HH, RM	0.1791			*	*	
HH, SD	0.05450			*		*
RM, SD	0.05579				*	*
BP, CI, HH	0.1312	*	*	*		
BP, CI, RM	0.1863	*	*		*	
BP, CI, SD	0.04894	*	*			*
BP, HH, RM	0.1381	*		*	*	
BP, HH, SD	0.04193	*		*		*
BP, RM, SD	0.04879	*			*	*
CI, HH, RM	0.1508		*	*	*	
CI, HH, SD	0.04504		*	*		*
CI, RM, SD	0.05604		*		*	*
HH, RM, SD	0.04525			*	*	*
BP, CI, HH, RM	0.1280	*	*	*	*	
BP, CI, HH, SD	0.03920	*	*	*		*
BP, CI, RM, SD	0.04982	*	*		*	*
BP, HH, RM, SD	0.03894	*		*	*	*
CI, HH, RM, SD	0.04672		*	*	*	*
BP, CI, HH, RM, SD	0.04111	*	*	*	*	*

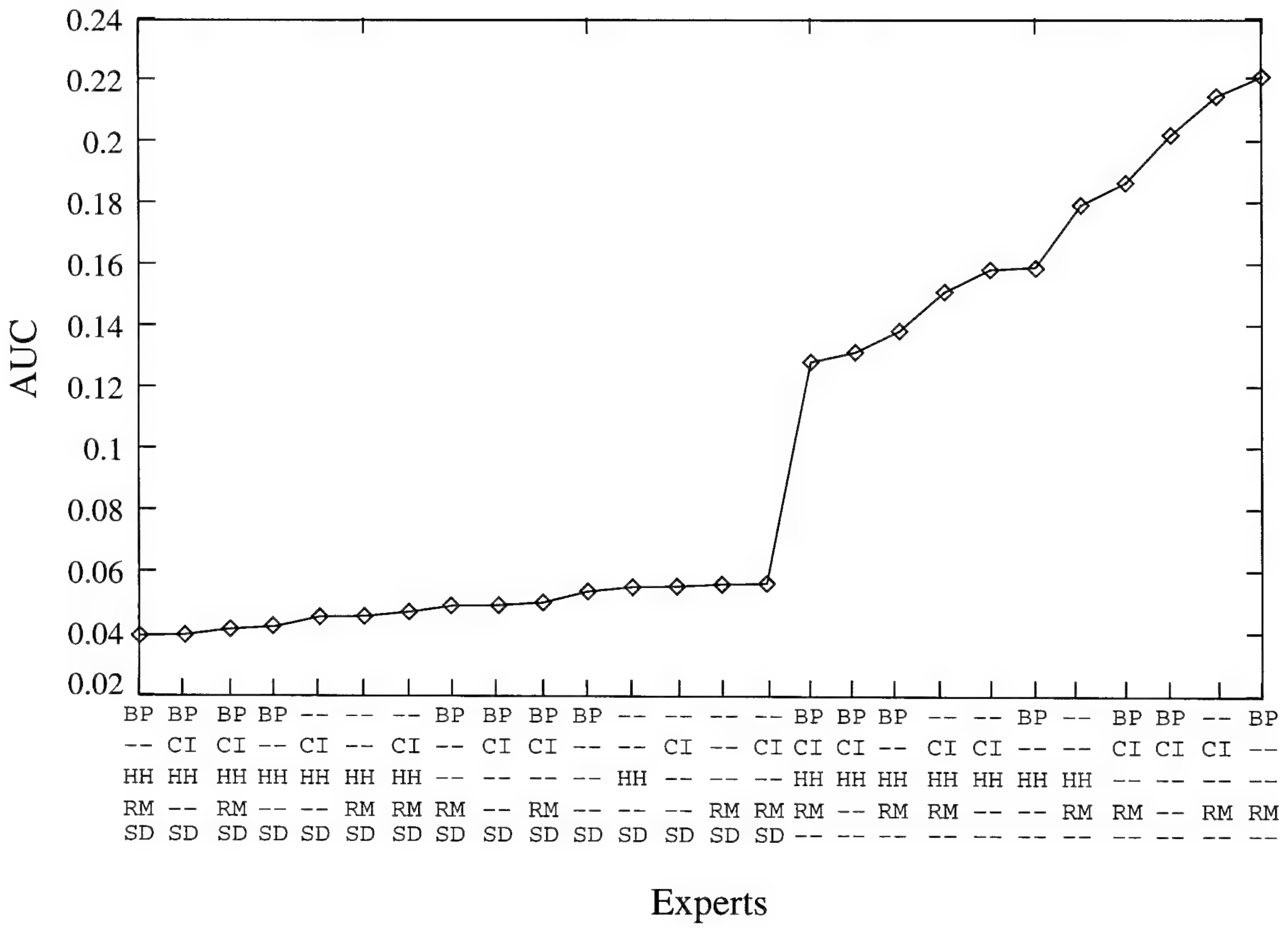


Figure 5.1: Discrimination provided by the Bayesian product rule. The 31 combinations of experts, where each combination is shown on the independent axis, are sorted from most to least discriminating.

Figure 5.2 shows the discrimination using each individual expert and all experts combined as the camera moves toward the target objects. Frame 0 is furthest from the target objects and frame 32 is closest. Each mean AUC value represents a single frame average across 50 tracking problems. Only the A block is tracked in these 50 problems from Block World Two, but the behavior is representative of all the frame-by-frame results.

Figure 5.2 supports three interesting observations. First, in every frame the combined estimate, using all five experts, is more discriminating than each of the individual experts. The statistical significance of the difference is tested using the same paired sample test and Bonferoni adjusted threshold used earlier. Only for frames 19 through 23 and 28 is the improved discrimination statistically ambiguous. Second, the figure shows stereo disparity is once again the most discriminating expert. Third, the discrimination of the Hausdorff-Huttenlocher expert appears to improve significantly on a frame-by-frame basis. This last observation deserves further investigation.

Figure 5.3 shows another set of mean AUC values partitioned by frame. For this figure, the mean AUC is shown for the color histogram back-projection (BP) and Hausdorff-Huttenlocher (HH) experts individually and combined. The target object is the C block in 10 Block World One tracking problems. Once again, the Hausdorff-Huttenlocher expert improves significantly as the frame index increases.

Considering the edge extraction process, the improvement of Hausdorff-Huttenlocher makes intuitive sense. The Hausdorff-Huttenlocher expert relies on the Canny edge detector to extract edge features. When the camera is far away from the target object, very few pixels represent the target object. When a target is represented with only a few pixels, the gradients used to indicate edge features are blurred and reduced, making edge features difficult to find [Can86]. For an extreme example, consider a target object represented by a single pixel — no gradients and no edges can be detected. As the camera moves closer to the target object, more pixels represent the target object, the edge gradients are more obvious, and the edge features are easier to find.

In comparison to Hausdorff-Huttenlocher, color histogram back-projection degrades only slightly with increasing frame index. The degradation is mostly due to the increasing size of the target objects. As the target objects get closer, the frame is filled with more and more of the target object color. As a result, back-projection identifies more and more of the pose space as containing the target object. In the limit, where the target object fills

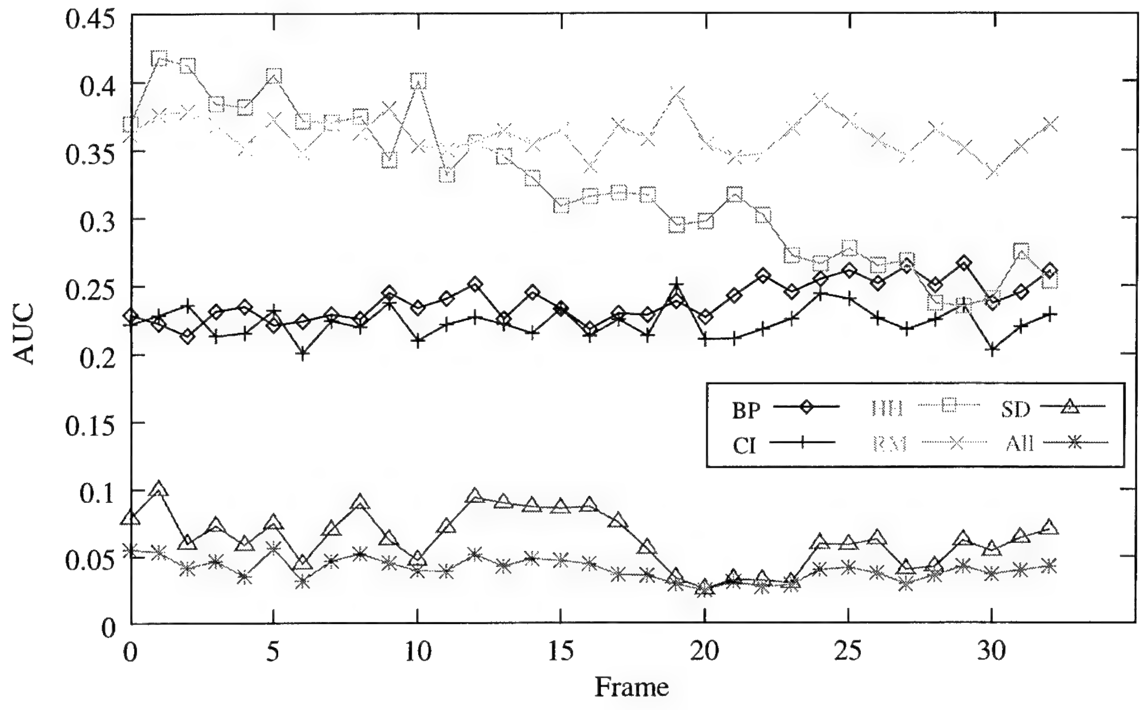


Figure 5.2: Comparing the discrimination using each expert individually (BP, CI, HH, RM, SD) to all the experts combined (All). Ten tracking problems in Block World Two are used to make the comparison, where the A block is the target object.

the camera's entire view, the entire pose space is identified as a probable match and no discrimination is provided. I have designed my problem environments so this limit is never reached: in the worst case closest frame the target object occupies only 23.2 percent of the image.

Walking through Figure 5.3 frame-by-frame illustrates several interesting points about the interaction of these two experts. First, near frame 0, when the Hausdorff-Huttenlocher expert provides very little discrimination, the combined estimate is very similar to back-projection alone. As Hausdorff-Huttenlocher becomes more discriminating, the discrimination of the combined estimate improves beyond the discrimination of stereo disparity alone. Eventually, the discrimination of the Hausdorff-Huttenlocher expert is better than the discrimination of back-projection alone. When back-projection is significantly less discriminating than Hausdorff-Huttenlocher, the discrimination of the combined expert begins to resemble Hausdorff-Huttenlocher alone. These results illustrate one way the Extended Condensation Filter combines estimates from experts as diverse as Hausdorff-Huttenlocher and color histogram back-projection.

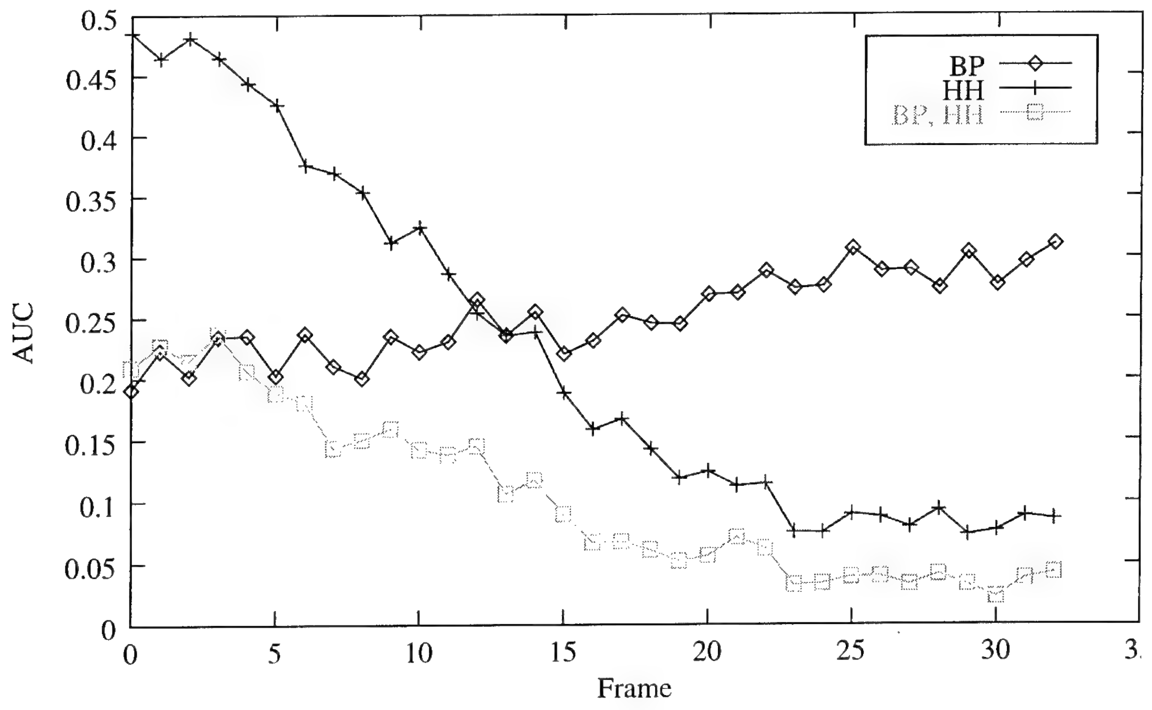


Figure 5.3: Comparing back-projection (BP) and Hausdorff-Huttenlocher (HH) discrimination. Ten tracking problems in Block World One are used to make the comparison, where the C block is the target object.

The results presented in Table 5.2, Figure 5.2, and Figure 5.3 indicate the Extended Condensation Filter combines expert probability estimates to improve discrimination. In general, all other partitionings of the results also support this conclusion. However, there are a few partitionings which provide statistically ambiguous results. There is also one specific case where this conclusion is not supported. The next two sections explore these exceptions to the generally favorable results.

5.1.3 Discrimination Ambiguity

Some of the data partitions provide ambiguous results for one of two reasons. First, some of the smallest partitions do not contain enough estimates to build an accurate ROC curve. For example, using only the estimates from a single frame of a single tracking problem in a single environment to build an ROC curve provides only 300 estimates. This small number of estimates creates an ROC curve where some points on the curve are estimated with very few samples. The resulting AUC estimates are very poor. When the AUC values are estimated using a small number of samples, the paired sample test indicates the difference between the individual experts and the combinations of experts is ambiguous.

Second, if two experts are combined and one expert provides a high level of discrimination, while the second expert provides a very low level of discrimination, the difference between the best expert and the combination may be very small. The first few frames of Figure 5.3 provide an example where Hausdorff-Huttenlocher provides very little discrimination, while color histogram back-projection provides much greater discrimination. The combined result is nearly the same as the discrimination provided by back-projection alone. In the first few frames, the combined discrimination is statistically indistinguishable from the discrimination provided by back-projection alone.

In the first case, the statistical testing is yielding ambiguous results as a consequence of limited resources. In other words, some partitions simply provide insufficient data. In the second case, the problem is more fundamental, but not troubling. In the limit, where a very good discriminator is combined with a relatively poor discriminator, the result should behave comparably to the good discriminator. Thus, the fact that no statistically significant improvement is observed for pairs of experts where one is relatively weak is entirely expected.

5.1.4 A Discrimination Anomaly

Among the large number of favorable results and a few ambiguous results, there is one set of conditions where the Extended Condensation Filter appears to fail to improve discrimination. As shown in Table 5.3, when only color index and render-match are combined for the tracking problems in either Block World One or Block World Two, the combined discrimination is worse than color index alone. Table 5.3 shows the AUC using only color index tracking the A block in Block World One is 0.1388. The discrimination using only render-match is 0.2408. The combined discrimination is 0.1503 — worse discrimination than color index alone. This anomaly is not present in the Toy World results.

Looking at a frame-by-frame partitioning of the color index and render-match results indicates a possible cause of the anomaly. Figure 5.4 shows the two experts respond in a very similar manner on a frame-by-frame basis. The similar performance of color index and render-match in Figure 5.4 is a stark contrast with the dissimilar responses of back-projection and Hausdorff-Huttenlocher in Figure 5.3. The frame-by-frame response of the two experts are clearly correlated.

Table 5.3: Discrimination using color index (CI) and render-match (RM) experts. Each mean AUC is the mean for 10 tracking problems AUC values. The last column indicates the combined mean AUC (CI,RM) is an improvement over both contributing experts (Yes), not an improvement (No), or statistically ambiguous.

		Mean AUC			
Environment	Target	CI	RM	CI, RM	Improvement
BW1	A	0.1388	0.2408	0.1503	No
	C	0.1816	0.2744	0.1943	No
	T	0.1856	0.2787	0.1950	No
BW2	A	0.2084	0.2776	0.2190	No
	C	0.2061	0.2988	0.2194	No
	T	0.2185	0.3520	0.2357	No
TW	Cat	0.3073	0.2670	0.2487	Yes
	Pig	0.3614	0.2704	0.2684	Ambiguous
	Rubik	0.3209	0.2167	0.1984	Ambiguous

To compare the correlation between color index and render-match to the correlation between other pairs of experts, the color index probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|w_{CI})$ and the render-match probability estimate $\hat{P}(\vec{S} = \vec{S}^{True}|w_{RM})$ are used to compute the Pearson correlation coefficient

$$r = \frac{\sum P_{CI}P_{RM} - \frac{\sum P_{CI} \sum P_{RM}}{N}}{\sqrt{(\sum P_{CI}^2 - \frac{(\sum P_{CI})^2}{N})(\sum P_{RM}^2 - \frac{(\sum P_{RM})^2}{N})}} \tag{5.2}$$

where P_{CI} and P_{RM} are shorthand for the conditional probability estimates, and N is the total number of pose estimates. The Pearson correlation coefficient is a commonly used measure of correlation. It measures the degree to which the pairs of conditional probability estimates fit a line — like simple linear regression [Hei96]. Tables 5.4, and 5.5 list the Pearson coefficient for each pair of experts using Block World One and Block World Two results. The tables indicate color index and render-match are consistently more correlated than any other pair of experts.

The high levels of correlation between color index and render-match could, in theory, cause the poor discrimination performance. The derivation of the Bayesian product rule in

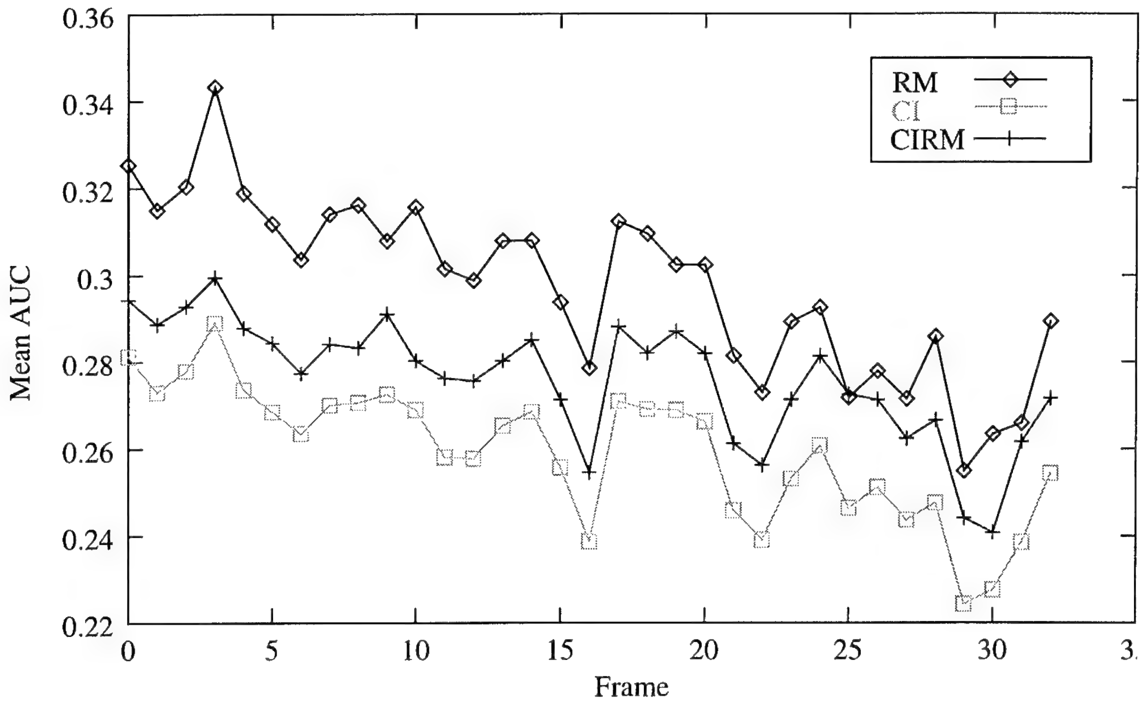


Figure 5.4: Comparing the discrimination using color index (CI), render-match (RM), and color-index and render-match combined (CIRM) on a frame-by-frame basis. Note the similar discrimination exhibited by the color index (CI) and render-match (RM) experts.

Chapter 3 assumes the experts are independent. The high correlation between color index and render-match indicates this key assumption is violated.

As shown in Table 5.3, the correlation of color index and back-projection is lower in Toy World than in either block world. In fact, the correlation is slightly lower than back-projection and render-match when they were successfully combined in Block World One. If high correlation is the source of the poor discrimination, the lower correlation between color index and back-projection in Toy World should improve the filter's performance. Table 5.3 shows the discrimination for color index and render-match in Toy World for each target object. When the pig or Rubik's cube are the target object, the filter appears to provide improvement, but the difference is statistically ambiguous. When the cat is the target object, the filter appears to provide an improved level of discrimination over either individual expert. The results in Toy World appear to match the prediction made from the correlation result.

The relationship between the anomaly and high-correlation is predicted by the derivation of the Bayesian product rule — providing a potential explanation for the anomaly.

Table 5.4: Pearson correlation (w_{RM}) between experts using only Block World One tracking problems.

Expert	BP	CI	HH	RM	SD
BP	1.00	0.672	0.154	0.701	0.472
CI		1.00	0.162	**0.806**	0.532
HH			1.00	0.139	0.497
RM				1.00	0.437
SD					1.00

Table 5.5: Pearson correlation (w_{RM}) between experts using only Block World Two tracking problems.

Expert	BP	CI	HH	RM	SD
BP	1.00	0.584	0.158	0.547	0.503
CI		1.00	0.1714	**0.764**	0.499
HH			1.00	0.0950	0.342
RM				1.00	0.366
SD					1.00

If, as the results indicate and Bayesian theory predicts, correlation can prevent improved discrimination, experts must be selected with some care. Alternatively, other expert combination rules may not react in the same manner to correlation between the experts. Results using the mean sum rule will be presented later in this chapter.

The way different experts interact with the combination rule may also explain why adding another expert to a combination may create worse results. As discussed earlier and shown in Figure 5.1, adding an additional expert to a combination of experts may increase or decrease the original discrimination provided by the combination. When two independent (uncorrelated) experts are combined using the Bayesian product rule, their combined discrimination is better than either contributing expert. If a third expert is added, which is highly correlated with one of the original two, this violation of the product rule assumption may degrade the result. I have not performed the tests necessary to support

Table 5.6: Pearson correlation (w_{RM}) between experts using only Toy World tracking problems.

Expert	BP	CI	HH	RM	SD
BP	1.00	0.550	0.0597	0.610	0.354
CI		1.00	0.0454	**0.696**	0.295
HH			1.00	0.102	0.252
RM				1.00	0.381
SD					1.00

this theory, but this is one possible explanation for the unpredictable results shown in Table 5.2 and Figure 5.1 when adding additional experts.

5.2 Accuracy and Precision

Intuitively, improved discrimination should lead to improved accuracy and precision. If an expert is better at discriminating between target and obfuscation, it seems it should be better at locating the target among the obfuscation. However, the scaled CEP results in this section indicate this may not be true.

As described in Section 4.2.2.1, the scaled CEP statistic provides a statistically normalized single measure of both the accuracy and precision in all dimensions of the pose space. In other words, statistically accurate and precise particle pose estimates, relative to the ground truth, must be present in all dimensions to create a low scaled CEP value. Therefore, I discuss scaled CEP as a measure of both accuracy and precision. To provide consistent results and the maximum time for convergence of the population, the state of the filter after the last frame of each tracking problem is always used to compute the scaled CEP values.

The results in this section support the following hypothesis originally posed in Section 4.2.2:

Hypothesis The CEP for the Extended Condensation Filter pose estimate using all the experts is less than the CEP using each individual expert.

Table 5.7: Comparing the scaled CEP using each expert individually and the experts combined. Each value is the mean scaled CEP using x and z only. All 450 training problems are used to create these results.

Expert(s)		BP	CI	HH	RM	SD
	Mean CEP	1.912	2.506	2.157	3.093	0.9133
BP	1.912		*	*	*	*
CI	2.506			*	*	*
HH	2.157				*	*
RM	3.093					*
SD	0.9133					
BP, CI, HH, RM, SD	0.5662	*	*	*	*	*

Only two exceptions are noted. First, highly correlated experts prevent an improved result. Second, the results in this section are unable to support or refute the ability of the filter to improve the orientation portion of the pose estimate because all of the experts failed to provide useful orientation information.

5.2.1 Training Problem Accuracy and Precision

Table 5.7 compares the accuracy and precision of the x and z portion of the individual expert pose estimates using the scaled CEP statistic. The second row shows the mean scaled CEP for each individual expert. The second column shows the mean scaled CEP for different combinations of experts. Due to the resources required to run 31 combinations of experts on 450 tracking problems and greater interest in the untrained problems, only one combination of experts is tested using the training problems. In the next section, all 31 combinations of experts are tested using the untrained problems.

Table 5.7 supports two interesting observations. First, the mean scaled CEP values indicate the combined estimate is more accurate and precise than each of the individual contributing experts. Second, comparing the scaled CEP values in Table 5.7 with the discrimination results in Table 5.2 indicates relative discrimination does not predict relative accuracy. Table 5.8 shows the discrimination rank and accuracy rank of each expert side-by-side. The number in parentheses is the AUC or CEP value. Ranking the individual measures

Table 5.8: Comparing discrimination to accuracy and precision for each expert. All the training problems are used to create these results.

Rank	Discrimination (AUC)	Accuracy (CEP)
1	SD (0.06855)	SD (0.9133)
2	CI (0.2365)	BP (1.912)
3	RM (0.2752)	HH (2.157)
4	BP (0.2872)	CI (2.506)
5	HH (0.2911)	RM (2.157)

by relative discrimination and accuracy shows these two properties correlate poorly. However, there are two patterns. First, stereo disparity provides the best discrimination and the best accuracy. Second, back-projection and Hausdorff-Huttenlocher together move ahead of color index and render-match.

The accuracy and precision results reported in Table 5.7 consider only the x and z portion of the target pose estimate. The next set of results consider only the orientation of the target object θ. These results are the only disappointing aspect of this research. In every case, the filter is unable to accurately determine the orientation of the target object. However, the failure is not due to a flaw in the filter, but my failure to implement a set of experts that provide useful orientation information.

5.2.2 Orientation Accuracy and Precision

Table 5.9 shows the orientation estimate error for each individual expert and all the experts combined. The orientation accuracy is measured using the mean median θ error. In other words, the median θ error is computed using all the estimates for one tracking problem, then these median θ errors are used to compute a mean error for all the tracking problems. In every case, as shown in Table 5.9, the mean median error is between 42 and 45 degrees. Considering the range of possible poses is only -90 to +90 degrees, an error of 45 degrees is extremely bad. In fact, the errors are about as good as guessing.

If a target object is oriented directly toward the camera, the ground truth orientation θ is 0 degrees. If a completely indiscriminant expert makes uniform random guesses at values between -90 and +90, the average error of the guesses will approach 45 degrees. In fact, the

Table 5.9: Comparing the accuracy and precision of orientation estimates using each expert individually and the experts combined. Each value is the mean scaled CEP using θ only. All the training problems are used to create these results. In every case the difference between experts is not statistically significant.

Expert(s)		BP	CI	HH	RM	SD
	θ Err (deg)	42.35	44.41	43.18	42.96	42.83
BP	42.35		94.8%	75.6%	67.4%	63.4%
CI	44.41			86.5%	87.2%	88.5%
HH	43.18				58.1%	61.7%
RM	42.96					54.0%
SD	42.83					
BP, CI, HH, RM, SD	44.43	91.1%	50.1%	80.3%	83.7%	82.5%

error for random guesses will always approach 45 degrees, regardless of the ground truth target orientation. The large orientation errors shown in Table 5.9 indicate the experts are about as good as guessing. They are able to discern little, if any, information about the orientation of the target object.

Four of the experts; back-projection, color index, Hausdorff-Huttenlocher, and stereo disparity, are not expected to provide much useful orientation information. Color histogram back-projection uses a color histogram to represent the target object, and rotation causes little change in the apparent color. Color index compares different instances of the rotated target, but it is dependent on color just like back-projection. Color is relatively rotation invariant. My implementation of Hausdorff-Huttenlocher only searches in translation and scale space, so a rotated object is largely unrecognizable. Stereo disparity only provides depth information and knows nothing about the target object.

In contrast, render-match was expected to provide useful orientation information. Render-match should be the most discriminating and accurate source of object rotation information. However, additional data indicates the x and z estimates may not be accurate enough to allow render-match, as I implemented it, to accurately determine orientation.

For the render-match algorithm, orientation estimates are very dependent on good x and z pose estimates. Figure 5.6 (a) plots the relationship between the render-match similarity metric (w_{RM}) and orientation error when the x portion of the pose estimate is perfect. The

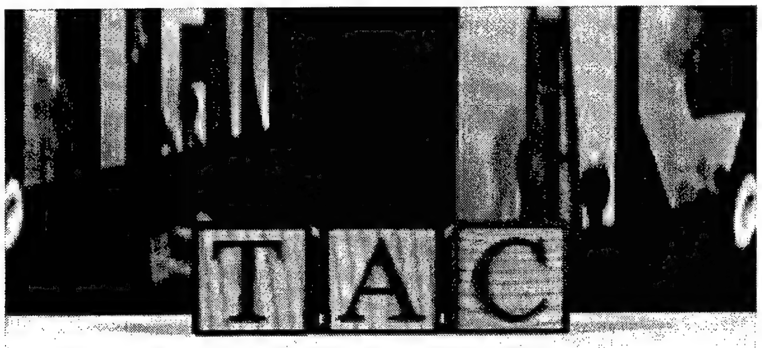


Figure 5.5: Unknown image used for the render-match correlation examples. The ground truth pose of the T block is $(x = -1.1, z = 7.0, \theta = 11.3)$

unknown image used to create this plot is shown in Figure 5.5 and the T block is the target object. From the camera point of view, the ground truth pose is (x=-1.1, z=7.0, θ=-11.3). The plot indicates there are two maximum correlation values. The left maximum is very near the ground truth orientation of 11.3 degrees. Using this data narrows the filter's search to two orientation values, providing useful information for the filter's search for the target orientation. However, a small amount of x pose estimate error dramatically changes this relationship.

Figure 5.6 (b) shows the relationship between w_{RM} when the x pose estimate is moved -0.1 units along the x axis from the ground truth. The ground truth orientation is no longer the maximum. Figure 5.6 (c) shows the relationship between w_{RM} when the x pose estimate is moved -0.2 units along the x axis from the ground truth. In this case, the global maxima are at the extreme orientations. In fact, the correlation when θ is less than -75 degrees is greater than the ground truth correlation. These figures indicate the relationship between θ error and correlation error inverts as x error increases — the global maxima are far away from the ground truth. This inversion may explain why render-match fails to provide accurate orientation estimates.

My implementation of render-match compares image patches to image patches. Only the small portion of the unknown image estimated to contain the target object is compared to the rendering of the target object template projected to the estimate $\vec{S}$. Chapter 3 describes this process in detail. Stevens predicted, and my results verify, the sensitivity of correlation when comparing image patches to image patches [SB01]. As an image patch is rotated, the number of pixels compared grows smaller. Smaller images tend to provide higher correlation, even if the image patches themselves are dissimilar. Instead of using image patches, Stevens compared images to images — ensuring the comparisons are always made using images of the same size.

Figure 5.7 shows the result of using an image to image version of the render-match algorithm on Figure 5.5. The single figure shows the correlation when the x pose estimate error is 0.0, 0.1, and 0.2. Searching the space represented by Figure 5.7 should provide superior results to searching the space represented by Figure 5.6 because the maximum correlation is in the neighborhood of the ground truth pose, even when large orientation errors and x and z pose errors are combined.

Unfortunately, comparing images requires significantly more computer time than comparing image patches because the entire image is always compared rather than a small part of the image. To reduce run-time, I elected to implement an image patch to image patch comparison. I hoped the interaction of the other experts would bring the pose estimates close enough to the ground truth to eliminate the problems caused by image patch to image patch comparison and allow my system to effectively employ far faster comparisons. Unfortunately, even with the contributions of the other experts, render-match still suffers from the image patch to image patch problem. As a result, my filter accurately estimates the x and z portions of the target pose, but fails to accurately estimate target orientation because no expert provides any useful information regarding orientation.

Despite the disappointing orientation results, this section indicates the Extended Condensation Filter, when tested on the training images, provides discrimination and pose estimates superior to any of the contributing experts, so long as the Bayesian independence assumption is met. However, the training data results in this section only measure the filter's ability to memorize responses to previously observed problems. The next results focus on the filter's ability to generalize to new problems.

5.2.3 Untrained Problem Accuracy and Precision

In this section, two tables and a figure illustrate the accuracy and precision of pose estimates for the problems withheld from training the filter — the untrained problems. Table 5.10

compares the individual experts to all possible combinations of experts. Figure 5.8 shows a second graphical view of the accuracy and precision results. Table 5.11 compares the performance of the filter on the training problems to the performance of the filter on the untrained problems.

Table 5.10 shows the accuracy and precision measured for all possible combinations of experts. The results are obtained with the same procedure used for the training problems. However, every combination of experts is tested. The CEP for each individual expert is shown across the tops of the columns. The rows contain each of the 31 possible combinations of experts. A "(+)" in the first column indicates the combined mean scaled CEP is an improvement over all of the contributing experts. A "(-)" in the first column indicates the combined mean AUC is worse than at least one of the contributing experts. Each value is the mean scaled CEP for the x and z portion of the pose estimate. Only the 90 untrained tracking problems are used to create these results.

Table 5.10 supports three interesting observations. First, stereo disparity provides the best accuracy and precision, despite the fact that it knows nothing about the target. Second, in nearly every case, the result of combining two experts improves the accuracy of the estimate provided by the individual experts. Only two combinations did not create a improved result — these combinations are marked with a "(-)". Third, the correlation between color index and render-match reported in an earlier section appears to affect the performance of the filter. Both combinations where accuracy is not improved include the color index and render-match operators.

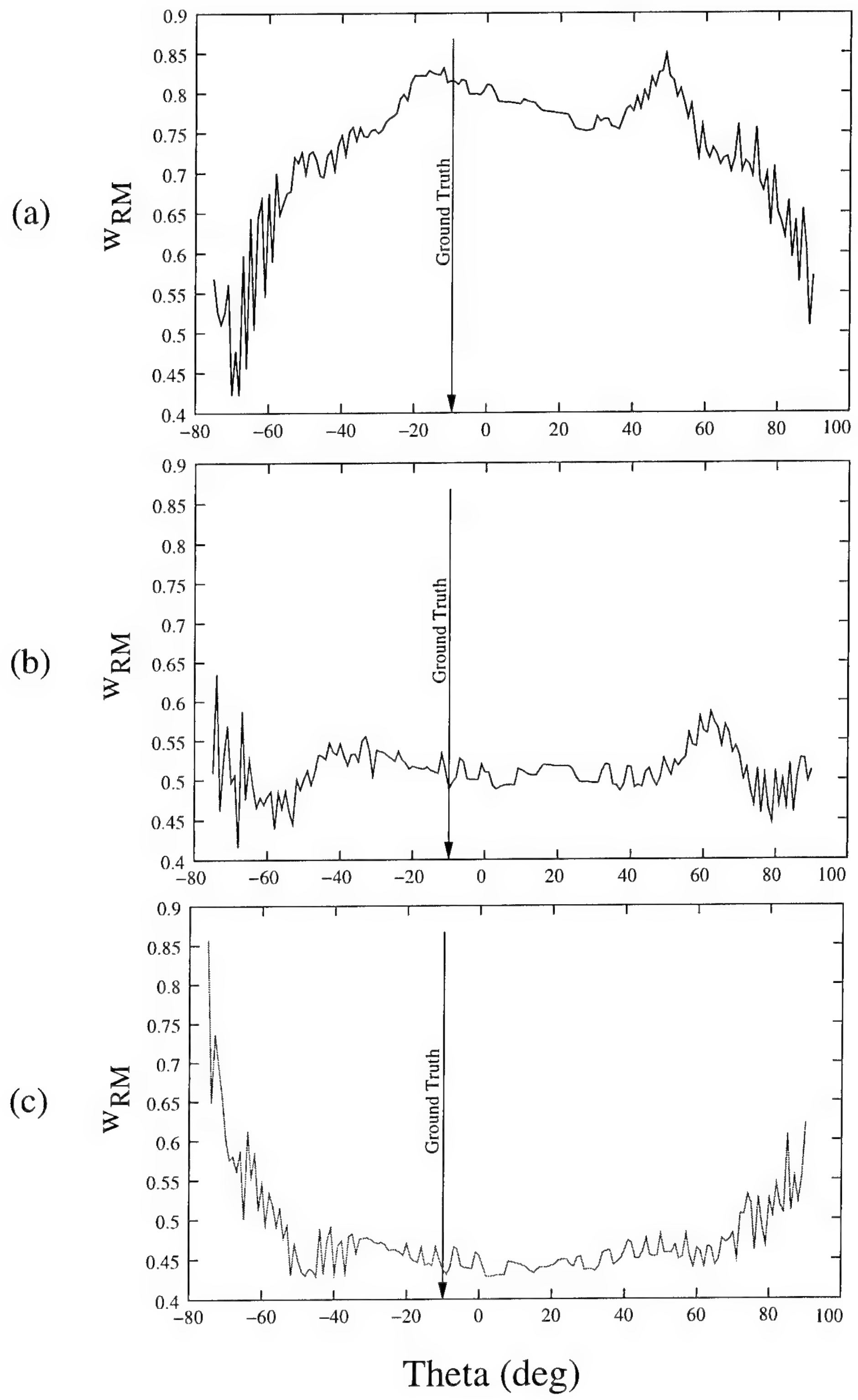


Figure 5.6: Example of image patch to image patch correlation (w_{RM}) as a function of orientation (θ) when the x pose estimate error is (a) zero, (b) 0.1, and (c) 0.2.

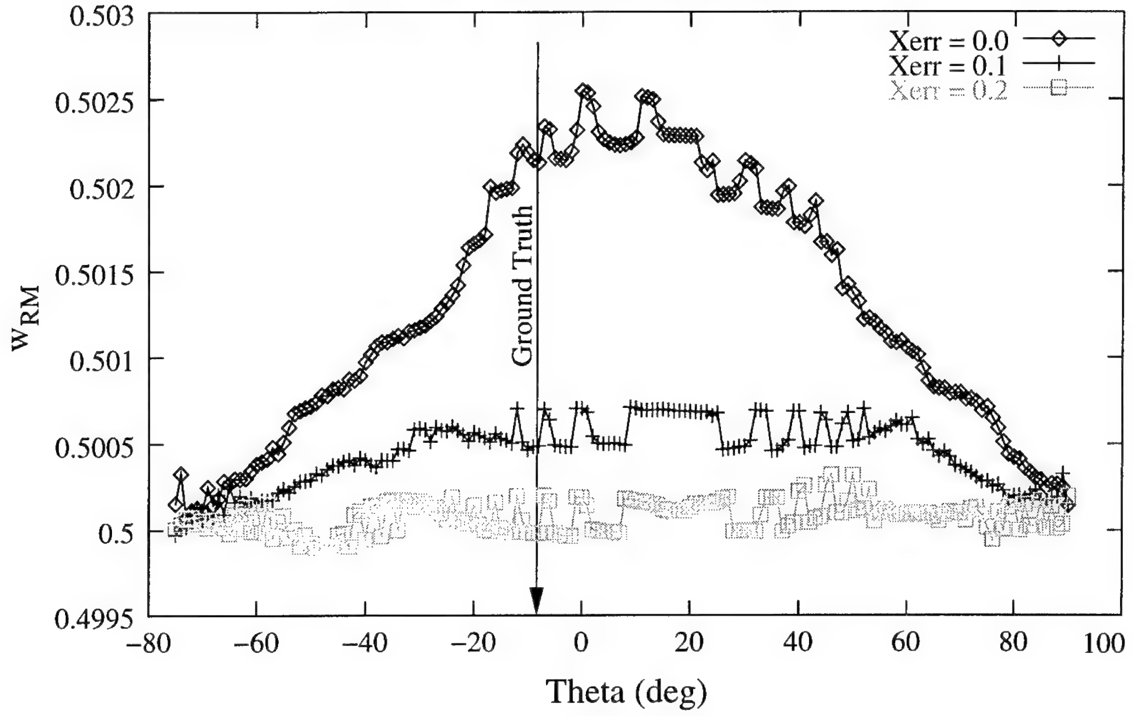


Figure 5.7: Example of image to image correlation as a function of orientation (θ) when the x pose estimate error is zero, 0.1, and 0.2.

Table 5.10: Comparing the accuracy and precision of pose estimates using different combinations of experts. The first five rows compare individual experts and the remaining rows compare combinations of two or more to the individual experts. All 90 untrained problems are used to create these results. Orientation θ is not included in these results.

Expert(s)		BP	CI	HH	RM	SD
	Mean CEP	2.093	2.601	2.267	3.256	0.9512
BP	2.093		*	98.0%	*	*
CI	2.601			*	*	*
HH	2.267				*	*
RM	3.256					*
SD	0.9512					
BP, CI (+)	2.012	*	*			
BP, HH (+)	1.981	*		*		
BP, RM (+)	1.808	*			*	
BP, SD (+)	0.8841	*				*
CI, HH (+)	2.208		*	*		
CI, RM (-)	2.936		*		*	
CI, SD (+)	0.9043		*			*
HH, RM (+)	2.177			*	*	
HH, SD (+)	0.8889			*		*
RM, SD	0.9310				*	97.3%
BP, CI, HH (+)	2.016	*	*	*		
BP, CI, RM (+)	1.964	*	*		*	
BP, CI, SD (+)	0.7037	*	*			*
BP, HH, RM (+)	2.001	*		*	*	
BP, HH, SD (+)	0.7658	*		*		*
BP, RM, SD (+)	0.6778	*			*	*
CI, HH, RM (-)	2.661		94.2%	*	*	
CI, HH, SD (+)	0.7046		*	*		*
CI, RM, SD (+)	0.7463		*		*	*
HH, RM, SD (+)	0.8697			*	*	*
BP, CI, HH, RM (+)	2.0813	*	*	*	*	
BP, CI, HH, SD (+)	0.7322	*	*	*		*
BP, CI, RM, SD (+)	0.6616	*	*		*	*
BP, HH, RM, SD (+)	0.7615	*		*	*	*
CI, HH, RM, SD (+)	0.8088		*	*	*	*
BP, CI, HH, RM, SD (+)	0.5968	*	*	*	*	*

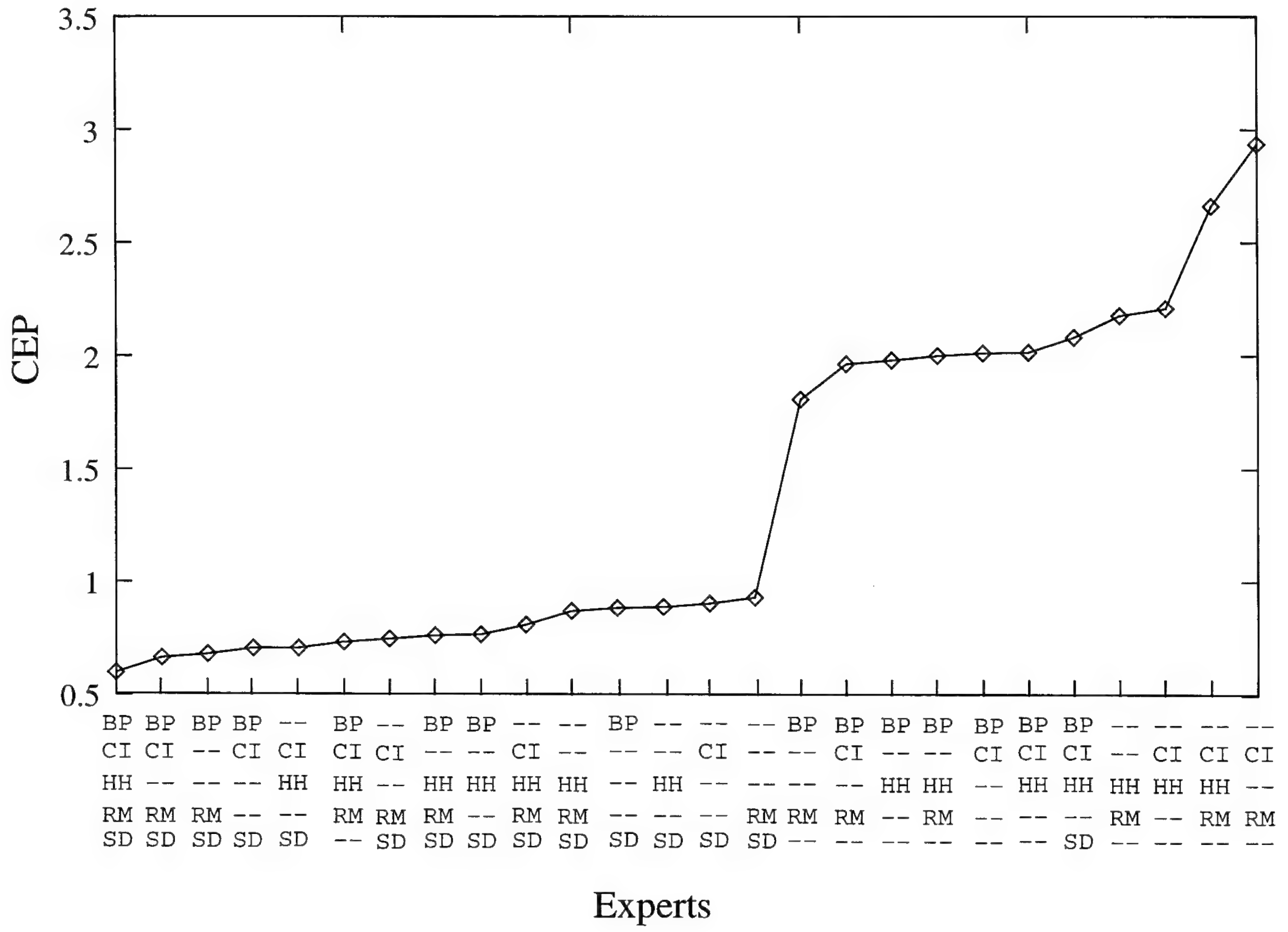


Figure 5.8: Accuracy and precision provided by the Bayesian product rule. Combinations of two or more experts are shown on the independent axis and sorted from smallest to largest CEP.

Table 5.11: Comparing the accuracy and precision of pose estimates using each expert individually. All 90 untrained problems are used to create these results. Orientation θ is not included in these results.

Expert(s)	Training Problems CEP	Untrained Problems CEP
BP	1.912	2.093
CI	2.506	2.601
HH	2.157	2.267
RM	3.093	3.256
SD	0.9133	0.9512
BP, CI, HH, RM, SD	0.5662	0.5968

Figure 5.8 provides a different view of the untrained problem results. Each of the combinations of experts is shown on the independent axis and sorted from most accurate and precise to least accurate and precise. The dependent axis shows the CEP for the combination. Once again, this figure shows stereo disparity usually dominates all the combinations of experts. However, there are two exceptions. The combination of back-projection, color index, Hausdorff-Huttenlocher, and render-match performs better than many of the combinations that include stereo disparity. The combination of back-projection, color-index, Hausdorff-Huttenlocher, and stereo disparity performs worse than many combinations that do not include stereo disparity. The discrimination results do not predict these two exceptions. Figure 5.8 will be used again later for a comparison to the filter's performance using the mean sum rule.

Table 5.11 compares the accuracy and precision of the filter on the training problems and untrained problems. As described earlier, only the individual experts and all the experts combined are tested using the training problems. Therefore, only those combinations can be compared. Two observations are supported by this table. First, the results obtained using the untrained problems are very similar to the results using the training problems. Second, despite similar relative performance of the experts, each expert's accuracy and precision is slightly worse when tested on the untrained problems as compared to the training problems. In general, the experts do generalize from their training examples, but generalization, as expected, is more difficult than memorization.

Table 5.12: Comparing the accuracy and precision of orientation estimates. Each error value is the mean median θ error. Using a 95% individual confidence threshold none of the mean values is statistically different. All 90 untrained problems are used to create these results.

Expert		BP	CI	HH	RM	SD
	Mean θ err (deg)	41.2	39.0	38.5	43.0	40.7
BP	41.2		77.2%	82.8%	73.7%	59.5%
CI	39.0			55.4%	90.5%	70.6%
HH	38.5				91.8%	74.0%
RM	43.0					80.0%
SD	40.7					

5.2.4 Untrained Problems Orientation Accuracy and Precision

Table 5.12 shows the orientation estimates for each expert. The results are similar to the orientation error using the training problems. In every case, the estimate error is very large and statistically indistinguishable from the other experts. Once again, it appears none of the experts provide any information about the orientation of the target object.

5.3 Likelihood – the Mean Sum Rule

To this point all the testing has used the Bayesian product rule to combine expert probability estimates. This section tests discrimination and accuracy and precision using the mean sum rule. Except for the substitution of the mean sum rule, the results in this section use the same analysis and presentation as the Bayesian product rule.

The mean sum rule computes the mean value of the expert probability estimates

$$L(\vec{S} = \vec{S}^{True} | \Delta_0, \ldots, \Delta_N) = \frac{\sum_{i=0}^{N} P(\vec{S} = \vec{S}^{True} | \Delta_i)}{N} \tag{5.3}$$

where Δ is one of the individual expert similarity measures and N is the number of experts. As discussed in Section 2.3.4, the mean sum rule does not have the strong theoretical support of the Bayesian product rule. However, the simple rule provides an interesting contrast to the Bayesian product rule and, in general, the results in this chapter indicate the mean sum rule provides better discrimination than the Bayesian product rule, but arguably worse accuracy and precision. The discrimination results also indicate the rule is not sensitive to high expert correlation.

The results in this section support the following hypothesis originally posed in Section 4.2.3:

Hypothesis Using the mean sum rule, the AUC for the combined estimate is less than the AUC for the estimate using each expert independently.

However, the results in this section frequently fail to support the following hypothesis originally posed in Section 4.2.3:

Hypothesis Using the mean sum rule, the CEP for the Extended Condensation Filter pose estimate using all the experts is less than the CEP using each individual expert.

The strong support for the first hypothesis and lack of support for the second hypothesis indicate, once again, that there is more to an accurate and precise estimate than good discrimination.

5.3.1 Discrimination using the Mean Sum Rule

Table 5.13 shows the discrimination for every combination of experts using the mean sum rule. The tables are created by partitioning the 90 untrained problem results by tracking problem and measuring the AUC for each problem. The mean for the 90 problems is shown. With the exception of only a few ambiguous results, as described in a Section 5.1.3, similar results are obtained using the other partitions.

A table comparing individual expert discrimination is not shown because the results are statistically indistinguishable from Table 5.2. The combination rule has no effect on the discrimination provided by a single expert.

Table 5.13: Comparing the discrimination provided by combining experts using the mean sum rule.

Expert(s)		BP	CI	HH	RM	SD
	Mean AUC	0.2872	0.2365	0.2911	0.2752	0.06855
BP, CI (+)	0.2005	*	*			
BP HH (+)	0.1583	*		*		
BP, RM (+)	0.2194	*			*	
BP, SD (+)	0.05597	*				*
CI, HH (+)	0.1532		*	*		
CI, RM (+)	0.2131		*		*	
CI, SD (+)	0.05219		*			*
HH, RM (+)	0.1583			*	*	
HH, SD (+)	0.05227			*		*
RM, SD (+)	0.05230				*	*
BP, CI, HH (+)	0.1243	*	*	*		
BP, CI, RM (+)	0.1838	*	*		*	
BP, CI, SD (+)	0.04690	*	*			*
BP, HH, RM (+)	0.1318	*		*	*	
BP, HH, SD (+)	0.03998	*		*		*
BP, RM, SD (+)	0.04757	*			*	*
CI, HH, RM (+)	0.1469		*	*	*	
CI, HH, SD (+)	0.03998		*	*		*
CI, RM, SD (+)	0.05126		*		*	*
HH, RM, SD (+)	0.04019			*	*	*
BP, CI, HH, RM (+)	0.1231	*	*	*	*	
BP, CI, HH, SD (+)	0.03483	*	*	*		*
BP, CI, RM, SD (+)	0.04647	*	*		*	*
BP, HH, RM, SD (+)	0.03507	*		*	*	*
CI, HH, RM, SD (+)	0.03894		*	*	*	*
BP, CI, HH, RM, SD (+)	0.03456	*	*	*	*	*

Table 5.14: Comparing the discrimination provided by the Bayesian product and mean sum rules. A "(+)" marks the better discrimination.

Experts	Product Rule Mean AUC	Mean Sum Rule Mean AUC	Confidence Level
BP, CI	0.2018	0.2005 (+)	*
BP HH	0.1589	0.1583 (+)	*
BP, RM	0.2208	0.2190 (+)	*
BP, SD	0.05345 (+)	0.05597	*
CI, HH	0.1581	0.1532 (+)	*
CI, RM	0.2144	0.2131 (+)	*
CI, SD	0.05511	0.05219 (+)	*
HH, RM	0.1791	0.1583 (+)	*
HH, SD	0.05490	0.05227 (+)	*
RM, SD	0.05579	0.05230 (+)	*
BP, CI, HH	0.1312	0.1243 (+)	*
BP, CI, RM	0.1863	0.1838 (+)	*
BP, CI, SD	0.04894	0.04690 (+)	*
BP, HH, RM	0.1381	0.1318 (+)	*
BP, HH, SD	0.04193	0.03998 (+)	*
BP, RM, SD	0.04879	0.04757 (+)	*
CI, HH, RM	0.1508	0.1469 (+)	*
CI, HH, SD	0.04504	0.03998 (+)	*
CI, RM, SD	0.05604	0.05126 (+)	*
HH, RM, SD	0.04525	0.04019 (+)	*
BP, CI, HH, RM	0.1280	0.1231 (+)	*
BP, CI, HH, SD	0.03920	0.03483 (+)	*
BP, CI, RM, SD	0.04982	0.04647 (+)	*
BP, HH, RM, SD	0.03894	0.03507 (+)	*
CI, HH, RM, SD	0.04672	0.03894 (+)	*
BP, CI, HH, RM, SD	0.04111	0.03456 (+)	*

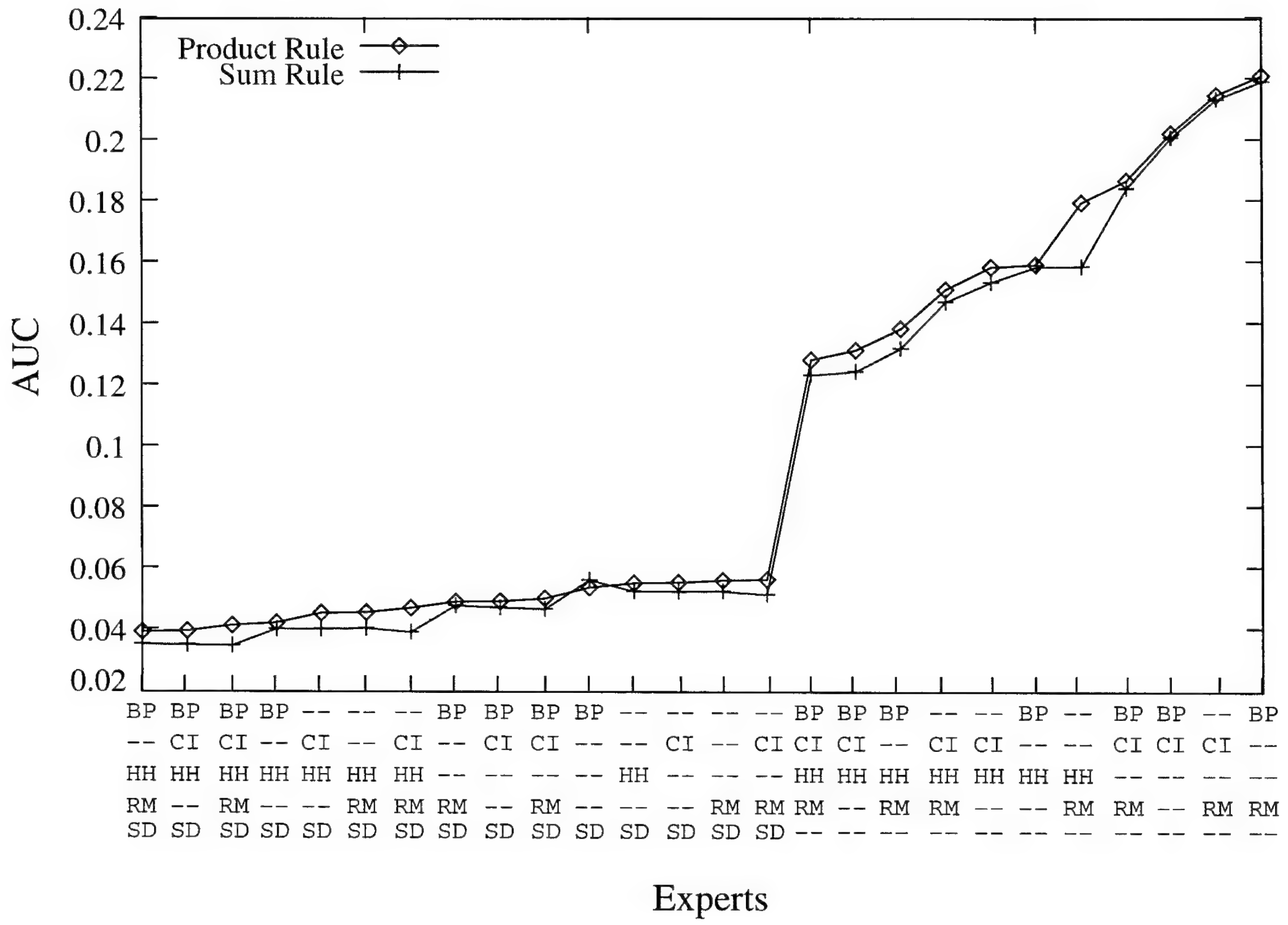


Figure 5.9: Comparing the discrimination provided by the Bayesian product and mean sum rules.

The results in Table 5.13 are similar to the results obtained using the Bayesian product rule. They indicate the combined discrimination using the mean sum rule is better than all of the contributing experts. In addition, the discrimination provided by the mean sum rule appears to be better than the Bayesian product rule, except when the experts color histogram back-projection and stereo disparity are combined (BP, SD). Table 5.14 tests the statistical significance of the improvement using the mean sum rule. In every case, the improvement provided by the mean sum rule is statistically significant.

Given the view of the mean sum rule as bagging, the superior discrimination provided by the mean sum rule indicates the experts are independent and unbiased. This is not too surprising considering the experts use entirely different methods — such as geometric versus radiometric or recognition versus classification.

Given Kittler's derivation of the mean sum rule, the superior discrimination is surprising. The context free assumption, that all experts are equally good under all conditions is obviously not true. For example, stereo disparity is far more discriminating than the other experts and, as Figure 5.3 shows, Hausdorff-Huttenlocher is clearly more discriminating when the camera is close to the target objects. As described in Section 2.3.4, Kittler's derivation of the mean sum rule also requires the conditional probabilities not deviate dramatically from the prior probabilities. Perhaps the experts in my implementation are "highly ambiguous due to high levels of noise" as Kittler's derivation requires [KHDM98].

The exception to the generally better discrimination provided by the mean sum rule is the combination of color histogram back-projection and stereo disparity. No explanation for this exception is apparent. So far as the Bayesian product rule is concerned, this pair of experts are not less correlated than all other pairs of experts, as shown in Tables 5.4, 5.5, and 5.6. However, it is possible back-projection and stereo disparity may, in some manner, violate the assumption of the mean sum rule more than any other combination of experts.

With only one exception, the mean sum rule provides superior discrimination. However, consistent with previous results, the superior discrimination provided by the mean sum rule does not necessarily create superior accuracy and precision.

5.3.2 Accuracy and Precision using the Mean Sum Rule

Table 5.15 show the results of testing the accuracy and precision provided by the mean sum rule on the 90 untrained problems. The format of the results is the same used earlier for the presentation of results using the Bayesian product rule.

The tables support several observations. First, accuracy and precision of the pose estimate are seldom improved. For six of 31 combinations of experts, the combined estimate does not not appear to provide an improvement. For 14 of the 31 combinations, the improvement is ambiguous. For only six of the 31 estimates is the combination an improvement over all the contributing experts. Second, as shown in Table 5.16, in some cases the accuracy and precision using the mean sum rule is better than the accuracy and precision using the Bayesian product rule. This last observation prompts a direct statistical comparison of the performance of the Bayesian product rule and the mean sum rule.

Table 5.15: Comparing the accuracy and precision of pose estimates provided by combining experts using the mean sum rule. Orientation θ is not included in these results.

Expert(s)		BP	CI	HH	RM	SD
	Mean CEP	2.093	2.601	2.267	3.256	0.9512
BP, CI	2.116	88.8%	*			
BP, HH	2.058	78.6%		*		
BP, RM	2.135	93.2%			*	
BP, SD	1.000	*				59.2%
CI, SD (+)	0.8527		*			*
CI, HH (-)	2.610		93.3%	*		
CI, RM (-)	3.002		*		*	
HH, RM (-)	3.186			*	94.2%	
HH, SD	0.9676			*	*	91.3%
RM, SD	1.053				*	78.0%
BP, CI, HH	2.093	80.0%	*	95.8%		
BP, CI, RM	2.529	*	62.5%		*	
BP, CI, SD (+)	0.7567	*	*			*
BP, HH, RM	2.098	86.8%		68.0%	*	
BP, HH, SD	0.9156	*		*		97.5%
BP, RM, SD (+)	0.8777	*			*	*
CI, HH, RM (-)	2.910		*	*	*	
CI, HH, SD	0.8866		*	*		94.9%
CI, RM, SD (-)	2.067		*		*	*
HH, RM, SD	0.9879			*	*	62.7%
BP, CI, HH, RM (-)	2.557	*	79.8%	*	*	
BP, CI, HH, SD (+)	0.7176	*	*	*		*
BP, CI, RM, SD (+)	0.8417	*	*		*	*
BP, HH, RM, SD (+)	0.8555	*		*	*	*
CI, HH, RM, SD	2.132		*	76.2%	*	*
BP, CI, HH, RM, SD	0.8890	*	*	*	*	95.9%

Table 5.16: Comparing the accuracy and precision provided by the Bayesian product and mean sum rules. A "(+)" marks the better result. Each value is the mean scaled CEP using x and z only.

Experts	Mean Scaled CEP		Confidence
	Product Rule	Mean Sum Rule	Level
BP, CI	2.012	2.116	84.3%
BP HH	1.981	2.058	76.6%
BP, RM	1.808	2.135	65.6%
BP, SD	0.8841	1.000	92.3%
CI, HH	2.208	2.610	50.7%
CI, RM	2.936	3.002	65.5%
CI, SD	0.9043	0.8527 (+)	*
HH, RM	2.177	3.186	80.2%
HH, SD	0.8889	0.9676	73.9%
RM, SD	0.9310 (+)	1.053	*
BP, CI, HH	2.016	2.093	91.0%
BP, CI, RM	1.964 (+)	2.529	*
BP, CI, SD	0.7037	0.7567	78.6%
BP, HH, RM	2.001	2.098	78.1%
BP, HH, SD	0.7658	0.9156	77.7%
BP, RM, SD	0.6778 (+)	0.8777	*
CI, HH, RM	2.661	2.910	60.6%
CI, HH, SD	0.7046 (+)	0.8866	*
CI, RM, SD	0.7463 (+)	2.067	*
HH, RM, SD	0.8697 (+)	0.9879	*
BP, CI, HH, SD	2.081	2.557	91.0%
BP, CI, HH, RM	0.7322	0.7176 (+)	*
BP, CI, RM, SD	0.6616 (+)	0.8417	*
BP, HH, RM, SD	0.7615	0.8555	92.7%
CI, HH, RM, SD	0.8088 (+)	2.132	*
BP, CI, HH, RM, SD	0.5968 (+)	0.8890	*

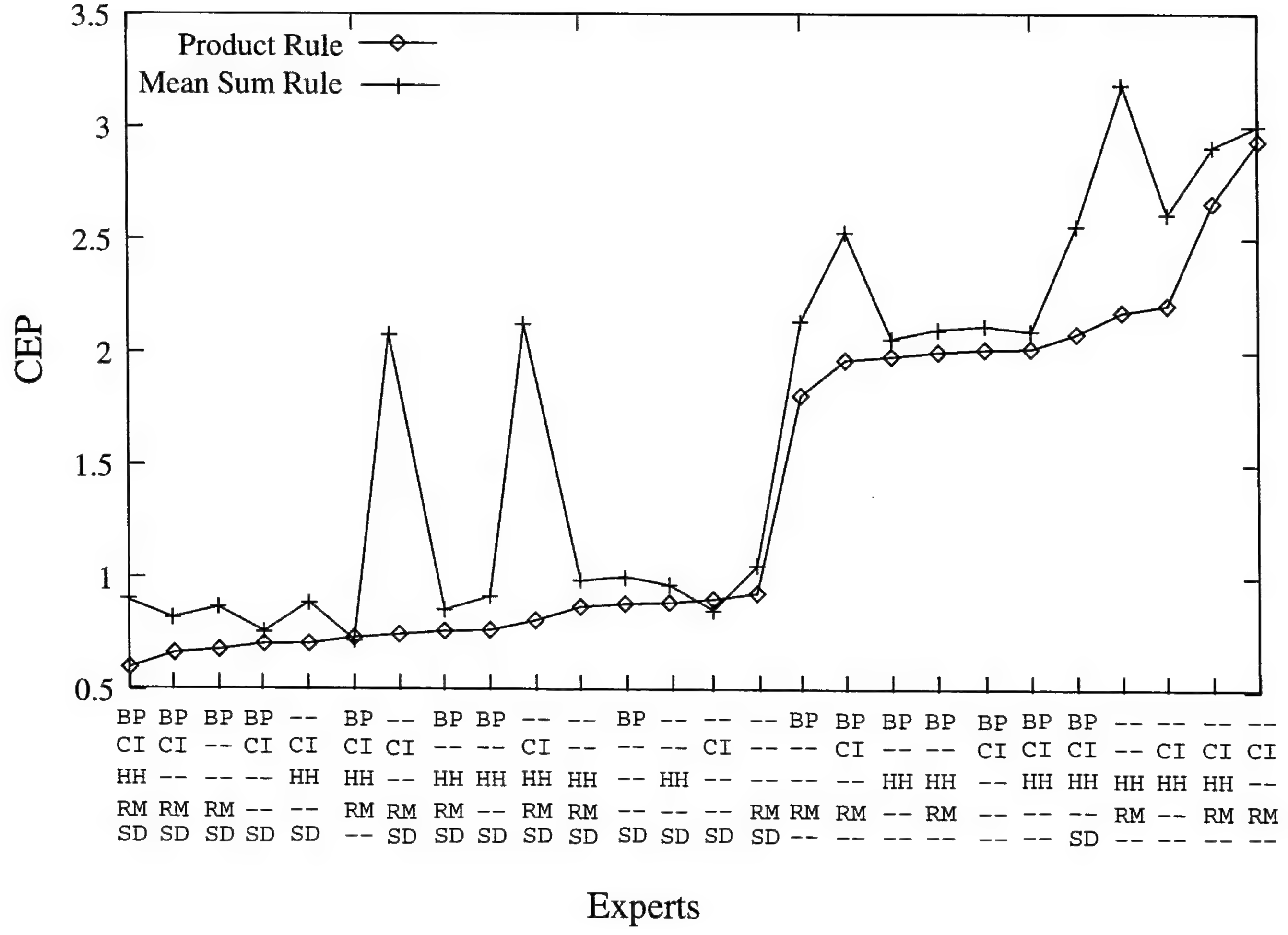


Figure 5.10: Comparing the accuracy and precision provided by the Bayesian product rule and mean sum rules. Each value is the mean scaled CEP using x and z only. The 31 combinations of experts, where each combination is shown on the independent axis, are sorted from lowest to highest CEP.

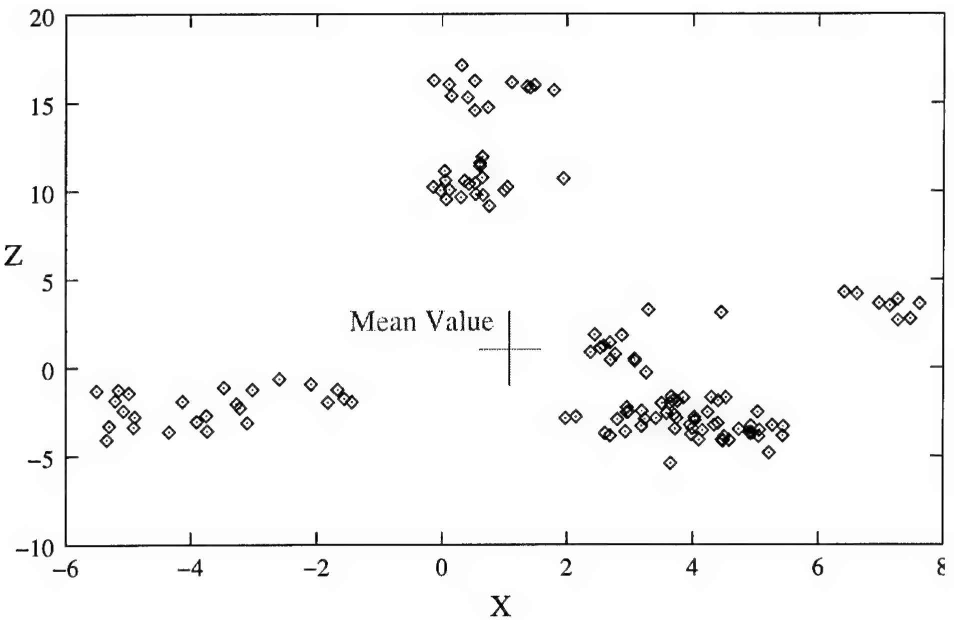


Figure 5.11: Example of multiple clusters in the x and z portion of pose space. This is the state of the filter after frame 32 for movie 1, target block A, Block World One training data.

Table 5.16 and Figure 5.10 compare the accuracy and precision using the mean sum rule to the accuracy and precision using the Bayesian product rule. The mean sum rule provides superior performance for two combinations, inferior performance for nine combinations, and ambiguous results for the other 15 combinations. These results are consistent with other partitionings of the data. In general, the Bayesian product rule provides better performance more often than the mean sum rule. This is a surprising result considering the mean sum rule provides better discrimination.

5.4 Convergence

The results presented so far purposely avoid assumptions about the distribution of particles within the filter. This section indicates there is good reason to avoid distribution assumptions. Figure 5.11 shows one example where using the mean value of all the particle poses to estimate the best pose is misleading — the mean pose is not well-supported by any of the particles. The problem is caused by the presence of multiple dense well-separated clusters in the filter. When multiple clusters are present, hypothesis extraction is more complex because each cluster must, in some manner, be identified.

The first test in this section conservatively counts the number of times multiple dense well-separated clusters are present in the filter. The clusters are counted after the last update using a hierarchical bottom-up clustering analysis, as described in the previous chapter. For the test, dense and well-separated is defined by the ratio of the mean difference to the sum of the standard deviations for two clusters

$$\frac{\|\mu_1 - \mu_2\|}{\sigma_1 + \sigma_2} \tag{5.4}$$

where μ_1 and μ_2 are the mean values for two clusters and σ_1 and σ_2 are the standard deviations for two clusters.

The results in this section do not support the following hypothesis originally posed in Section 4.2.5:

Hypothesis The entire population of the Extended Condensation Filter is reduceable to a single statistically similar cluster using a bottom-up hierarchical clustering algorithm.

In addition to the presence of multiple clusters, the results indicate a second compelling reason for analyzing the behavior of clusters in the filter. As noted in Chapter 3, authors frequently claim the configuration of the Condensation Filter particles represents the conditional probability of the state space [D+99, GSS93, Gor97, IB96, IB98], although a question about the validity of the associated proofs was raised. In other words, for my application, the particles should provide an estimate of the conditional probability $P(\vec{S} = \vec{S}^{True})$. If so, when the filter is presented with two equally good hypotheses, the filter should maintain two clusters of points representing two hypotheses.

A second set of test in the next section indicate, given enough updates, the population eventually converges to a single cluster, even when two equally good target pose hypotheses are present. Therefore, the configuration of particles in the filter do not, at least in my test problems, appear to represent the underlying conditional probability.

5.4.1 Convergence in the Tracking Problems

Tables 5.17 and 5.18 present histograms counting the number of dense well-separated clusters in the filter after frame 32 of a tracking problem. Table 5.17 counts the number of clusters after the last frame of the 450 training problems. Table 5.18 counts the number of clusters after the last frame of the 90 untrained tracking problems. For all the cluster tests,

Table 5.17: Histograms counting the number of times 1, 2, 3, or 4 clusters are present after the last frame of all 450 training problems. The counts are obtained using hierarchical bottom-up clustering.

Environment	Target	Cluster Count 1	2	3	4
BW1	A	37	12	1	
	C	39	9	2	
	T	42	7	1	
	All	118	28	4	
BW2	A	32	13	5	
	C	37	8	5	
	T	36	10	2	2
	All	105	31	12	2
TW	C	39	7	4	1
	P	34	9	6	
	R	37	6	7	
	All	110	22	17	1
All Training		333	81	33	3

the filter is run with all of the expert probability estimates combined using the Bayesian product rule.

The first two columns of each table show the problem environment and object tracked, the third column shows the number of times one dense well-separated cluster is present in the filter, the fourth column shows the number of times two clusters are present, and so on. An intermediate total for each problem environment and a final total for all trained or untrained problems is also provided.

The methods used to derive the tables are designed to conservatively under-count the number of dense well-separated clusters present in the filter. Multi-dimensional clustering is a notoriously difficult problem [Mur85]. Even humans looking at the same arrangement of data points may disagree on the number of clusters present. To avoid these problems and provide a conservative estimate of the number of clusters present, I perform clustering

Table 5.18: Histograms counting the number of clusters present after the last frame of all 90 untrained problems. The counts are obtained using hierarchical bottom-up clustering.

		Cluster Count					
Environment	Target	1	2	3	4	5	6
BW1	A	6	3	1			
	C	10					
	T	9	1				
	All	25	4	1			
BW2	A	6	4				
	C	4	3	2			1
	T	8	1	1			
	All	18	8	3			1
TW	C	6	1	3			
	P	8	1	1			
	R	6	3	1			
	All	20	5	5			
All Untrained		63	17	9			1

in only one of the three dimensions at a time — the x dimension is shown in the tables. As a result, clusters that may be dense and well separated in two dimensions are sometimes projected and flattened into a single cluster.

To provide conservative cluster counts, I also apply equation 5.4 with a very conservative maximum threshold of 3.0. As a result, my clustering algorithm may occasionally combine clusters that visually appear dense and well-separated. For example, two Gaussian clusters, one with ($\mu_1 = -3, \sigma_1 = 1$) and the other with ($\mu_1 = 3, \sigma_1 = 1$) sit right at the 3.0 threshold, but visually they appear dense and well-separated. Move the means slightly further apart and they are counted as two clusters. Move the means slightly closer together and they are counted as one cluster.

The conservative counts in Tables 5.17 and 5.18 support two observations. First, after 32 frames of a tracking problem, the filter usually contains a single cluster. Second, multiple clusters are not uncommon. Based on the number of times multiple dense well-separated clusters are present in the filter and the conservatism of these counts, simply computing the mean particle pose may mis-represent the state of the filter.

5.4.2 Cluster Dynamics

Tables 5.17 and 5.18 show the number of clusters in the filter after frame 32, but a different view helps illustrate the dynamic behavior of the clusters. Figure 5.12 shows the typical evolution of particle clusters on a frame-by-frame basis. For the figure, only the z dimension of each particle is shown. Negative z values are possible because the coordinate system is world relative rather than camera relative. At frame 0, the particles are initialized in a uniformly random distribution between $z = 1$ and $z = 20$. As the frames are used to update the filter, the particles condense to two well supported areas of the pose space. The results are from movie one in Block World One, where the C block is the target object and only the render-match expert is used.

On average, Figure 5.12 appears to contain fewer and fewer clusters after each frame. Even at the very end of the problem, a small cluster of points at approximately $z = -7$ merges with a larger cluster. This result is typical of the behavior in other dimensions and other tracking problems. This led to the question, does the population always condense to a single cluster? For example, given enough updates, will the two clusters shown in Figure 5.12 eventually combine into a single cluster?

Given infinite computer resources, I would run all 540 tracking problems until either a single cluster is present or the population appeared static. In the absence of these resources, I elected to perform a limited test using a worst case problem.

5.4.3 Always a Single Cluster — Eventually

If the Extended Condensation Filter represents the conditional probability $P(\vec{S} = \vec{S}^{True})$ and two equally good pose estimates, $\vec{S}_1$ and $\vec{S}_2$ are present, where

$$P(\vec{S}_1 = \vec{S}^{True}) = P(\vec{S}_2 = \vec{S}^{True}) \tag{5.5}$$

The filter should maintain two dense well-separated clusters. This theory suggests a test where the filter is provided with a movie containing two equally good hypotheses and run to convergence. The results in this section indicate, even in the presence of two identical instances of the target, two dense well separated clusters are not maintained.

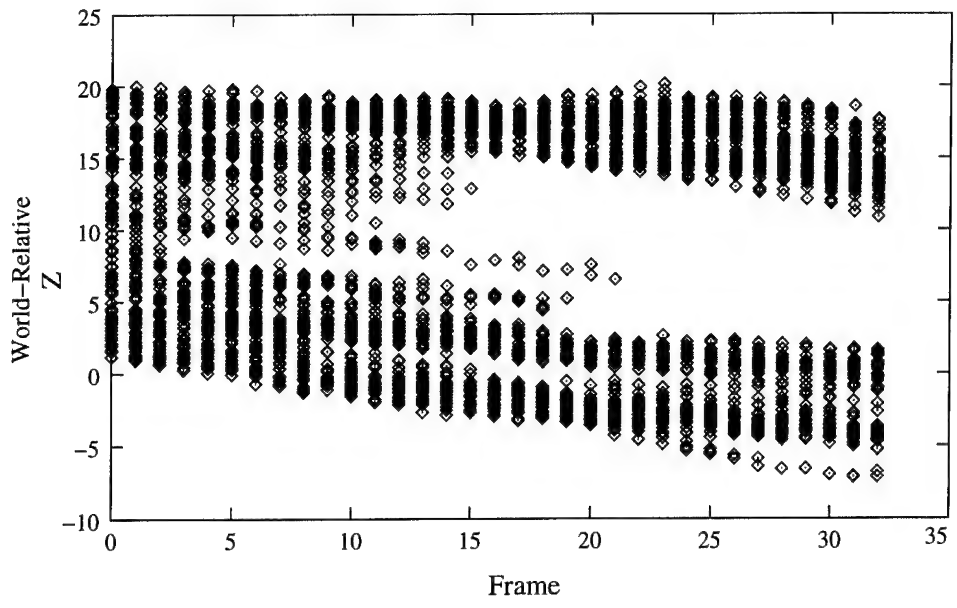


Figure 5.12: Dynamic behavior of clusters. Each point is the world-relative z coordinate of a particle's pose estimate after a frame is processed.

Figure 5.13 shows the only unknown image I_t used for this set of tests. The test image is constructed by repeating the same 160x240 target image twice to create one 320x240 target image — the left and right sides of the image contain identical pixels. As a result, the experts all respond identically to the two halves of the unknown test image. Therefore, the underlying conditional probability is identical at the left and right target pose and identical in the vicinity of the left and right pose. Due to the complications involved with making perfectly equal target poses in both left and right stereo disparity images, the stereo disparity expert is not used. Only the color histogram back-projection, color index, Hausdorff-Huttenlocher, and render-match experts are used.

To test the ability of the filter to model the two identical conditional probability densities created by my test image, I ran the filter until it converged on 30 different tests. For each test, the filter is randomly initialized in exactly the same manner as the block world and Toy World problem environments — the same number of particles and initial pose space. The Block World Two training data is used. The only two differences between the block world tests and this special test are the input image I_t, shown in Figure 5.13, and the movement

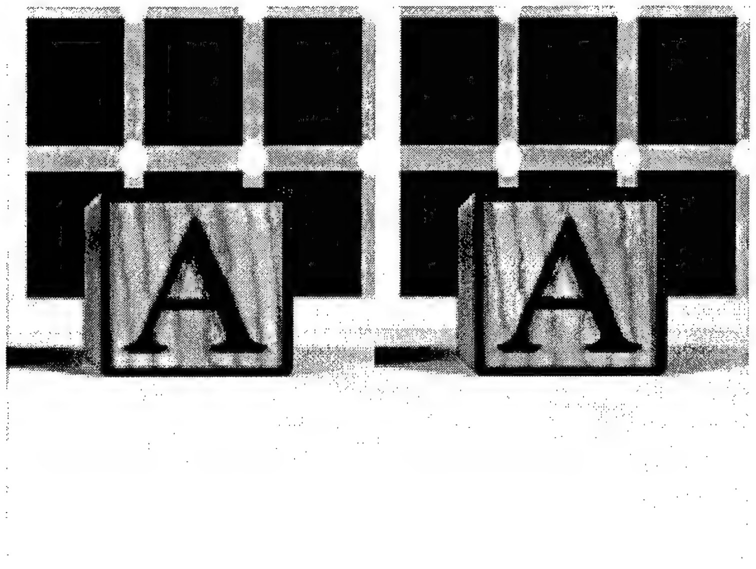


Figure 5.13: Image used to test worst case convergence.

of the camera. The motion model is changed to include only the Gaussian random variable.

$$\begin{bmatrix} x_{i+1} \\ z_{i+1} \\ \rho_{i+1} \end{bmatrix} = \begin{bmatrix} x_i \\ z_i \\ \rho_i \end{bmatrix} + \begin{bmatrix} Gaussian(0,1) \\ Gaussian(0,1) \\ Gaussian(0,1) \end{bmatrix} \tag{5.6}$$

where ρ is the camera ground-plane rotation about the y-axis, or "yaw" in aircraft terminology. This model represents a motionless camera.

To test for convergence, I used the same bottom-up hierarchical clustering test described at the beginning of this section, but I changed the convergence threshold from 3.0 to 1.0 for these tests because, in contrast to the previous clustering test, I want to err by failing to combine clusters that should be combined. I do not want to mistakenly signal there is a single cluster in the filter, when multiple clusters are still present.

For the 30 tests, the mean number of frames required to converge to a single cluster is approximately 130.2 with standard deviation of approximately 108.0. In the shortest case, the filter converged almost immediately at seven frames. In the longest case, the filter required 370 frames to converge to a single cluster. The probability of converging to the left target pose is approximately 0.43 and the probability of converging to the right target pose is approximately 0.57. These results indicate the filter always converges to a single hypothesis, even when the underlying conditional probability contains two equally good hypotheses.

A technique frequently required for statistical filter implementations may dramatically affect these results. My implementation of the Extended Condensation Filter equates frame rate with iteration rate. In other words, the update steps are only performed once per new unknown image. This is not necessary. The update steps could easily be performed multiple times for each new unknown image. In my experience, multiple update cycles per measurement are commonly used for Kalman Filter implementations. This may be required when experts provide asynchronous estimates— different experts provide information at different frequencies. Intuitively, given multiple iterations of the filter per new image, it seems the Condensation Filter population should converge to a unimodal population faster. However, this intuition is not tested in this dissertation.

It is possible the extensions accelerate the convergence of the filter population. When the extensions are used with the Condensation Filter, more particles near one of the two instances of the target causes more thorough searching near that pose. More thorough searching increases the probability of finding the optimal pose. Finding the optimal pose will create higher particle weights. Therefore, as more particles move near one instance of the particle pose, the weights associated with that cluster of particles will probably increase just because search is more thorough. This possibility is a consequence of the more active role played by the filter given the new extensions.

Population convergence is not considering the theory of random selection. As shown in Chapter 3, a fixed-size population of points with a static probability of selection, the population will eventually converge to a single member. As explained in Chapter 3, every time the population is reproduced there is a finite probability a member will not be selected. If the member is not selected, it is eliminated from the population. Adding noise to the motion model, which acts like a mutation operator, complicates the convergence issue. However, one would expect the population to converge to the neighborhood defined by the mutation function — this more complicated deserves additional study in future research.

5.5 Run Time

When developing a meta-system, such as the Extended Condensation Filter, which relies on input from other algorithms, the relative time allotted to algorithms and their relative contribution is often interesting. For example, is the most discriminating expert also the

Table 5.19: Wall clock run time required for each expert per movie frame. The 2,880 iterations of the filter for the Block World One, Block World Two, and Toy World untrained problems were used to create these statistics.

Expert	μ (mSec)	σ (mSec)
Canny Edge Feature Extractor	203.5	16.8
Hausdorff-Huttenlocher	1,277.6	231.8
Color Index	25,305.7	727.9
Render-Match	25,242.9	702.3
Color Index and Render-Match	26,719.3	1,065.4
Color Histogram Back-Projection	36.3	3.1
Stereo Disparity	1,027.3	160.1

most time consuming expert? In this chapter I have dedicated extensive resources to measuring the contribution, in terms of discrimination and accuracy and precision, of the five experts. However, little has been said about the time required for each expert.

Table 5.19 shows the approximate mean wall clock run time and standard deviation for each expert per movie frame using a 600MHz Intel Pentium III. The table should not be misconstrued as an attempt to compare the efficiency of various algorithms — my implementation of these algorithms is by no means a standard. The results merely represent the mean wall clock times I observed during my experiments.

Several observations are supported by the table. First, the most discriminating expert is not the most time consuming. Stereo disparity demonstrated the best discrimination, but requires less run time than all but the color histogram back-projection algorithm. Second, color index and render-match are shown separately and together to illustrate the time consumed by the process of rendering the target image at the pose $\vec{S}$ and the time required by the classifiers. The rendering step is time consuming, but it is only required once for all the classifiers. As a result, one classifier requires the time consuming process of rendering the target image, but additional classifiers require little additional time. This is illustrated by the fact that color index and render-match running together require little more time than running either algorithm alone.

5.6 Summary

The results in this section indicate the Extended Condensation Filter is capable of combining the diverse information provided by recognition, classification, and low-level features to discriminate and localize better than each of the individual contributing experts. The filter combines and propagates probabilistic representations of this information, so crisp decisions and the resulting loss of information are avoided. The results also question the indicate the Condensation Filter does not represent the conditional probability, instead it has much in common with search algorithms, such as Evolution Strategies.

5.6.1 Research Questions

In addition to the general observations above, the results in this chapter also support more specific observations regarding the research and implementation questions posed in Chapter 4. In this section, each of these questions is repeated and answered in light of the results in this chapter.

Research Question How does the discrimination of the combined probability estimate compare to the discrimination of the probability estimate from each of the experts?

The results in sections 5.1.1 and 5.1.2 indicate, with only one exception, the Extended Condensation Filter's combined probability estimate is more discriminating than the estimate from each of the individual contributing experts. This is true whether the Bayesian product rule or the mean sum rule is used. The only exception occurs when two highly correlated experts are combined using the Bayesian product rule.

Research Question How do the accuracy and precision of the Extended Condensation Filter's combined estimate compare to the accuracy of the estimate from each of the experts?

The results in section 5.2.1 indicate, with only one exception, the Extended Condensation Filter's combined estimate, when using the Bayesian product rule, appears to be more accurate and precise than estimates from the individual contributing experts. Once again, the exception occurs when two highly-correlated experts are combined.

Research Question Does the Extended Condensation Filter's learning algorithm generalize sufficiently to support operation in new portions of the problem domain?

The results in sections 5.2.3 and 5.3.2 indicate the Extended Condensation Filter generalizes well from the training examples. The untrained problems only caused a small degradation relative to the performance of the filter using the training problems. The filter also continues to provide more discriminating, accurate, and precise estimates than any of the contributing experts when the untrained problems are used.

Research Question Is the weight for each particle required to be a probability estimate or, as Pearl indicates, is relative likelihood sufficient to weight the Extended Condensation Filter's particles?

If discrimination and localization are viewed as equally important, neither the mean sum rule nor the Bayesian product rule appear to have a clear advantage. The results in Figure 5.9 indicate the mean sum rule may provide superior discrimination, when compared to the Bayesian product rule. In all but one case, the mean sum rule provides better discrimination. In one case, where correlation between experts prevents improved discrimination using the Bayesian product rule, the mean sum rule still provides improved discrimination. The results in Figure 5.10 indicate the Bayesian product rule may provide slightly better accuracy and precision. The Bayesian product rule provided a more accurate and precise estimate more often than the mean sum rule. However, the results are somewhat ambiguous.

The results point to another conclusion — discrimination alone does not necessarily result in improved accuracy and precision. The observations that the mean sum rule provides slightly better discrimination, while the Bayesian product rule provides slightly better accuracy and precision provides additional insight to Pearl's observations discussed in Section 2.3. Pearl indicates likelihood is required to decide where to expend resources. For the Extended Condensation Filter, likelihood is required to decide where to place more samples and intensify the target pose search. However, the filter's particles can only represent likelihood for a small number of poses, relative to the entire pose space. They cannot, as I believe Pearl assumes, represent the likelihood for all possible outcomes — the likelihood of all possible poses in my case. Therefore, like any search algorithm, the filter can get stuck in local maxima — the most likely pose currently visible within the neighborhood of all the particles.

Therefore, for the Extended Condensation Filter, likelihood may not be sufficient. Because the search algorithm embedded within the Extended Condensation Filter relies on gradient information, the derivative of the likelihood function is just as important as the individual likelihood values themselves. In other words, if increased discrimination is achieved at the cost of increasing the number of local minima, the increased discrimination may indirectly make the localization problem more difficult and decrease the accuracy and precision of the filter's estimate.

Research Question What are the convergence properties of the filter population? Specifically, how common is a multi-mode distribution of filter particles?

The results in section 5.4.1 indicate a user of the Condensation Filter should not assume the filter contains a single hypotheses and, for example, use the mean value to extract a hypothesis. However, it appears, given enough updates, the filter will eventually converge to a single hypothesis. A user needs, in some manner, to test for this condition.

The results in section 5.4.3 indicate the filter inevitably convergences to a single hypothesis. This result does not agree with the common assumption that a Condensation Filter represents the conditional probability associated with the state space. The results show the Condensation Filter will eventually converge to a single dense cluster of particles, even if the conditional probability associated with the state space contains two equally probable pose hypotheses.

5.6.2 Implementation Questions

In addition to suggesting answers to the research questions, the results provided in this chapter suggest some answers for the implementation questions posed in Chapter 4. In this section, each of these questions is repeated and answered in light of the results in this chapter.

Implementation Question Does the Extended Condensation Filter implementation degrade gracefully in the presence of objects with geometry poorly modeled by a single polygon?

Toy World is identical to Block World Two, except for the substitution of less-rectilinear target objects. In comparison to Block World Two's rectilinear alphabet blocks, Toy World uses the Rubik's cube, fuzzy cat, and pig, which are poorly modeled by a single polygon. The Rubik's cube template views the cube edge on. The cat is round, self-occluding and covered with fur. The pig is partly made of round surfaces and self-occluding. These objects are poorly modeled by a single two-dimensional template.

The color index and render-match experts rely heavily on the projection of the two-dimensional model into pose space. Despite the difference between Block World Two and Toy World, the performance of color index and render-match in Block World Two and Toy World is not dramatically different. Table 5.3 shows both experts provide similar levels of discrimination in both Block World Two and Toy World. The filter failed to determine the orientation of the target objects, but the failure occurred in both block worlds and Toy World. Nothing indicates either expert failed to accurately and precisely determine the target object x and z pose due to the two-dimensional representation of the Toy World targets.

Investigation of the discrimination anomaly indicates the filter performs better in Toy World than either block world. Correlation between two experts appears to prevent improved discrimination using the Bayesian product rule in the block worlds, but an undetermined property of Toy World reduced the correlation between the two experts.

Implementation Question Does color provide the best source of discrimination in the synthetic problem environments?

Discrimination results throughout this chapter indicate stereo disparity consistently provides the best discrimination for all the problems. In some cases the discrimination is four to five times better than any other expert. Based on the relative performance of stereo disparity, range information, rather than color, is the most discriminating source of information in the three problem environments.

The Hausdorff-Huttenlocher expert is completely unaware of the color content of any image. The results in section 5.1.2 show the Hausdorff-Huttenlocher expert provides almost no discrimination early in the tracking problems, when the camera is far away from the target objects. Later in the tracking problems, when the camera is closer to the target objects, it appears to perform better than color-based experts like color histogram back-projection. In the later frames of the tracking problems, geometry, rather than color appears to provide better discrimination.

In general, the Condensation Filter extensions proposed earlier in this dissertation have proven their value on a large set of randomly generated tracking problems. The filter has demonstrated the ability to combine extremely diverse sources of visual information to improve the discrimination and accuracy and precision of object recognition. However, the performance was not flawless. Specifically, the render-match algorithm should be re-implemented to compare images to images rather than patches to patches so the filter's ability to improve orientation estimates can be tested. Several questions remain open as well. For example, the results in this chapter are based entirely on synthetic imagery. Are the results representative of the filter's performance using physical sensors? This and other areas for future research are discussed in the next chapter.

Chapter 6

Conclusion

The results in the previous chapter indicate the Extended Condensation Filter achieves the goals stated at the beginning of Chapter 3. First, to guarantee generality, the Extended Condensation Filter performs information fusion without relying on specific parametric representations. Look-up tables are used to model conditional probabilities for each expert and a population of particles is used to represent the combination of multiple expert estimates.

Second, the state of the Extended Condensation Filter population is guided entirely by probabilistic representations. Pose estimates, similarity measures, and low-level features are converted to estimates of the conditional probability of a target pose. At every stage of the filter's operation, probabilistic representations of the information from the previous stage are carried forward, minimizing the loss of information that typically occurs with crisp decisions.

Third, pose estimates, similarity measures and low-level features are accommodated by the Extended Condensation Filter by converting them all to estimates of the conditional target pose probability. Object recognition algorithm pose estimates are accommodated in the manner used by the canonical Condensation Filter, except that the probability of the true target pose, conditioned on the distance to a recognition algorithm pose estimate, is learned. Similarity metrics provided by classification algorithms are accommodated using the pose estimate associated with each particle to provide a point of comparison. Low-level features provided by feature extraction algorithms are used to modify the conditional probabilities and narrow the space of probable target pose estimates.

In addition to satisfying the original goals, the test results also provide some surprises. First, despite the claims of previous authors, the population of the Condensation Filter does not in some cases represent the underlying conditional probability. There are also some questions about the proofs that the Condensation Filter represents the conditional probability. In some cases the Condensation Filter appears to find a maxima, like a population-based search.

If the Condensation Filter does behave similarly to population-based search, specifically an Evolution Strategy, this could be considered good news. A wealth of Evolution Strategy literature offers many potential improvements for the Condensation Filter. For example, the canonical Condensation Filter tends to converge to a single cluster of particles. In applications where multiple targets are known to exist in the pose space this behavior may be undesirable. Several papers propose kin competition or the niche operator to force diversity into the population [Gol89, Fog00]. Adding such an operator to the Condensation Filter could force the population to maintain multiple clusters and represent multiple pose estimates.

Evolution Strategy research has also focused on learning and estimating the optimal exploration strategy, such as estimating the covariance of the underlying fitness function in the attempt to tune mutation [Fog00]. If adapted to the Condensation Filter, on-line estimation of the uncertainty functions might eliminate the need for some *a priori* uncertainty models. The uncertainty might even be modified as the filter operates to improve performance as the recognition problem changes or in different parts of the pose space. The similarity between Evolution Strategies and the Condensation Filter provides many other opportunities to exploit Evolution Strategy research to benefit the Condensation Filter.

Classifier combination research is another area with the potential to improve the Condensation Filter. The Bayesian product rule and mean sum rule are only two of many combination rules. Others include the weighted sum rule, max rule, min rule, and median rule [KHDM98]. In addition to these simple rules, many adaptive techniques might be applied, such as graph models, reinforcement learning, and variations of the neural network [Hin99]. Each of these methods for combining expert estimates contains specific assumptions about the contributing experts. These assumptions affect the combined result, just as my results indicate the independence assumption affects the product rule and the context independence assumption affects the mean sum rule. Other combination rules may provide improved results depending on the information sources and problem domain.

Another surprising result is the orthogonality of discrimination and localization. The discrimination measured in Chapter 5 using the AUC statistic has little correlation with the accuracy and precision measured using the CEP statistic. Intuitively, it would seem the ability to discriminate between the target object and obfuscation would lead to more accurate localization of the target object. However, the data indicates this is not the case and recent results in cognitive science indicate our own visual system may have the same property.

The search model of the Condensation Filter may provide a practical explanation of why discrimination and localization are orthogonal properties within the Condensation Filter. As described in the AUC description in Chapter 3, greater discrimination means greater separation between the target and non-target distributions. Therefore, greater discrimination equates to greater difference, on average, between target and non-target probability estimates. However, many other properties of the fitness function make search easy or difficult. Factors such as the number of local minima and gradient information affect the performance of a search algorithm [MW95]. Discrimination does not measure these properties and, as a result, cannot estimate the search problem difficulty or the localization problem difficulty. Therefore, discrimination cannot predict the difficulty of the search problem or the accuracy and precision provided by a discrimination algorithm's similarity metric.

This dissertation indicates the Extended Condensation Filter can improve object recognition. However, the results also create as many questions as they answer. For example, the results are based entirely on synthetic imagery. Are the results representative of the filter's performance using physical sensors? This question is the obvious and appropriate next step for this research. In addition, the exploitation of Evolution Strategy research, the exploration of new combination rules, and the investigation of the relationship between discrimination, accuracy, and precision are just a few of the other more obvious areas for future work.

Appendix A

Mathematical Symbols

This appendix includes a list of the mathematical symbols used throughout this dissertation.

BP = subscript indicating the Color Histogram Back-Projection low-level feature extraction algorithm

CI = subscript indicating the Color Index classification algorithm

$d_{\vec{s}}$ = distance between object recognition pose estimate $\vec{s}$ and particle's pose $\vec{S}$

Δ = a generic symbol representing object recognition related information. This information could be a target pose estimate $(\vec{s}, w_i)$, a similarity metric w_i, or a low-level feature $\vec{f}$.

$(\Delta_0, \ldots, \Delta_N)$ = indicates a collection of sources of information, potentially including target pose estimates, similarity metrics, or a low-level features. See Δ above.

H_R RGB histogram for the input image

H_T RGB histogram for the target image

HH = subscript indicating the Hausdorff-Huttenlocher object recognition algorithm

i = filter particle index

I = an image

I_{EX} = image of the ground truth target object

I_0 = first unknown image acquired

I_t = an unknown image acquired at time t

I_T = last unknown image acquired

j = feature vector index

J = total number of feature vectors extracted for a single image

$L(A|B,C)$ = mean value of the probabilities $P(A|B)$ and $P(A|C)$

$M_{\vec{S}}$ = matrix used to project an image to the point $\vec{S}$ in pose space

M_{cm} = vector representing the camera's own motion in camera-relative coordinates

M_{tm} = vector representing the target motion in camera-relative coordinates

n = index of information sources

N = total number of particles in the Condensation Filter

$P(A|B)$ = the probability of event A, given event B has occurred

$\hat{P}(A|B)$ = an estimate of the probability of of event A, given event B has occurred

phi = vertical rotation axis around the camera-relative z-axis

RM = subscript indicating the Render-Match classification algorithm

σ_{cm} = the uncertainty (standard deviation) used in the camera motion model

$\vec{S}$ = a vector in pose space coordinates, usually the pose associated with a particle in the Condensation Filter population

$\vec{S}^{True}$ = the ground truth pose of the target object, a vector in pose space coordinates

S_x^{True} = horizontal translation x component of the ground truth pose of the target object

S_z^{True} = depth translation z component of the ground truth pose of the target object

S_θ^{True} = rotation θ component of the ground truth pose of the target object

SD = subscript indicating the Stereo Disparity low-level feature extraction algorithm

T = total number of unknown images acquired

$theta$ = horizontal rotation axis around the camera-relative y-axis

rho = vertical in-plane rotation axis around the camera-relative x-axis

u = down to up axis on the image plane

v = left to right axis on the image plane

$v_x, v_y, v_z, v_\phi, v_\theta, v_\rho$ = velocities

$v_x(\sigma,\mu), v_y(\sigma,\mu), v_z(\sigma,\mu), v_\phi(\sigma,\mu), v_\theta(\sigma,\mu), v_\rho(\sigma,\mu)$ = velocities defined as random variables

$\mu_{v_x}, \mu_{v_y}, \mu_{v_z}, \mu_{v_\phi}, \mu_{v_\theta}, \mu_{v_\rho}$ = mean velocities

$v_x, v_y, v_z, v_\phi, v_\theta, v_\rho$ = velocities

w = a classifier similarity score for a specific algorithm, such as the pixel correlation w_{RM} used for the render-match algorithm.

x = left to right axis in camera-relative pose space (left-hand coordinate system)

y = down to up axis in camera-relative pose space (left-hand coordinate system)

z = rear to forward axis in camera-relative pose space (left-hand coordinate system)

Appendix B

POVRay Synthetic Test Environment

Figure B.1 illustrates the use of a Python program and the Persistence of Vision Ray-tracer (POVRay) to render more than 11,880 photo-realistic images used to test the extended condensation filter. POVRay version 3.1g is an open source photo-realistic rendering system available from www.povray.org. POVRay combines most of the features found in the best commercial ray-tracing systems with a text-based modeling language, rather than a graphical user interface. The text-based programming language allows me to write model files and initiate rendering from a controlling Python program. Python version 1.5.2 is an open source interpreted language available from www.python.org. Python was used to write the model and ground truth files, control rendering, and store the results in a movie database.

Each of the three problem domains, such as Toy World, are represented by 60 movies. As shown in Figure B.1, three sets of files are needed to render each movie — the camera motion model, the world model, and the object models. The camera motion model is a small set of user-defined parameters including the camera's initial starting point and six-dimensional velocity. For my dissertation all the velocity components are zero except for a positive velocity along the z-axis. The world model is randomly generated by the Python program as described in Chapter 4. The object models, such as the alphabet blocks, pig, or cat, include the geometry and textures for each target object.

The Python script uses the randomly generated camera motion model, world model, and object models to build an image specific pair of scene models. One scene model is used for the left stereo camera and one scene model is used for the right stereo camera. The scene

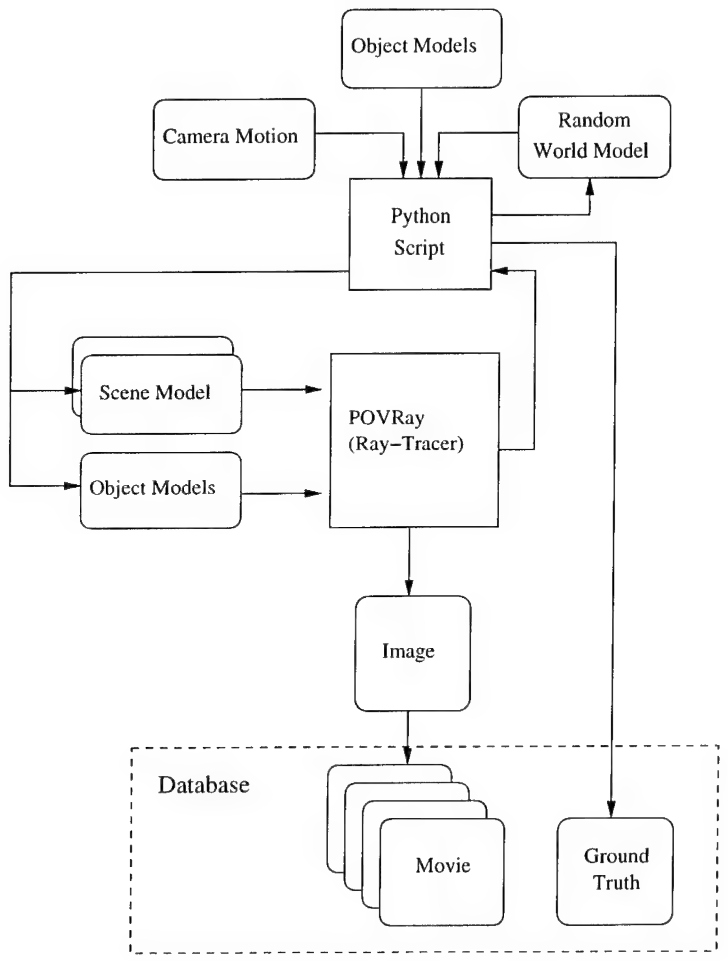


Figure B.1: Creation of movies (sequential images) and target object ground truth.

models and object models are passed to POVRay for rendering. Each scene model represent one of the 33 unique camera positions rendered for each movie. With each increment of time, the new position of the camera is computed, a new set of scene models are written and rendered, then the images are stored in the movie database and the ground truth file is updated. When the movie is complete, the ground truth file contains the camera motion at each time increment and the world-relative position of every target object. This ground truth information defines the value of $\S^{True}$ used during training and testing.

POVRay has the ability to produce image sequences — movies. However, to precisely control the target positions, render two images per camera position, and simplify file storage, I manually control POVRay using Python. Manually controlling POVRay also allows me to distribute rendering across multiple computers by giving different computers on the department's network a sub-set of the 66 images required to make a single movie.

All of my test imagery is rendered using the same rendering configuration. POVRay's quality control parameter is set to the highest non-experimental level — level nine. Level nine computes diffuse, reflected, refracted and transmitted rays. Level nine also also computes shadows. However, level nine does not include the radiosity algorithms which are considered experimental in the 3.1g version of POVRay. In addition to the maximum non-experimental quality level, recursive adaptive anti-aliasing with a 30% error threshold is used. The POVRay anti-aliasing process is recursive and adaptive because the rays are sub-divided until the pixel color estimated by the sub-set of rays is consistent to within 30%. As few as four rays or as many as 289 rays may be used to estimate each pixel's color — the anti-alias maximum number of recursive divisions is set to four.

POVRay offers additional features that could increase the realism of the imagery, such as radiosity, more sophisticated lens models, and atmospheric effects. However, additional realism incurs the risk of using experimental software and costs significantly more computer time. Considering over 11,880 images were rendered, the process described represents a good trade-off between realism and the computer resources required.

Appendix C

Non-Parametric Paired Sample Test

C.1 Basic Non-Parametric Paired Sample Test

The paired sample randomization test, as described in [Coh95], provides a distribution independent method for comparing the performance of two systems tested under identical circumstances. Using identical tests for two systems allows the variance of test conditions to be eliminated. Using the non-parametric randomization process also requires no distribution assumption.

For my evaluation, different combinations of experts are tested using either 150 trained tracking problems or 30 untrained tracking problems. One combination of experts is evaluated on a set of tracking problems and the circular error probable (CEP) is computed for each problem. A second combination of experts is evaluated on the exact same set of problems and the CEP statistics are computed. The goal is determining the statistical significance of the difference in the CEP values recorded for each combination of experts.

Table C.1, provides a small hypothetical example using two combinations of experts and six tracking problems. Each combination of experts is tested on the six tracking problems. The six tracking problems are created from two movies with three objects each. The difference between the CEP values for each combination of experts is shown in the last column. The mean difference between the combinations of experts is $\overline{x}_\delta$. In my example $\overline{x}_\delta$ happens to equal 0.67.

Table C.1: Example non-parametric paired sample t-test data

Tracking Problem	Combo. 1 CEP	Combo. 2 CEP	δ
Movie 1, Target A	2	2	0
Movie 1, Target C	7	8	1
Movie 1, Target T	5	4	1
Movie 1, Target A	4	3	1
Movie 1, Target C	4	4	0
Movie 1, Target T	5	4	1
		$\overline{x}_\delta$	0.67

To test the statistical significance of this difference, the null hypothesis $H_0 : \overline{x}_\delta = 0$ is used. In other words, I will test the hypothesis that the mean difference between the two combinations of experts is 0. A traditional paired sample t-test could be used at this point. However, a distribution of CEP values is not Gaussian. The minimum CEP value is 0 and the distribution is asymmetric among other potential problems. Therefore, I use the non-parametric paired sample test which makes no assumption about the sample distribution.

The null hypothesis $H_0 : \overline{x}_\delta = 0$ asserts the sample CEP values are from the same distribution. If the two populations of samples are from the same distribution, randomly swapping half of the CEP scores obtained from one combination of experts with CEP scores obtained from the other combination of experts should produce a sample of random mean difference values where half the values are above the observed mean difference and half the values are below. To the extent this does not happen, the null hypothesis is refuted. This is similar to a traditional t-test, in that the randomly produced mean differences create a test distribution serving the same role as the t-distribution. The further out on the tail of this test distribution the observed mean difference lies, the greater the statistical significance of the observed mean difference.

Table C.2 shows the example CEP data again, but the values are randomly swapped with probability 0.5. In this trial, the first, second, and fifth values values happen to be swapped and, as a result, $\overline{x}_\delta = 0.33$. In this trial, the new value is less than the observed

Table C.2: One random swapping trial

Tracking Problem	Combo. 1 CEP	Combo. 2 CEP	δ
Movie 1, Target A	2	2	0 (swap)
Movie 1, Target C	7	8	-1 (swap)
Movie 1, Target T	5	4	1
Movie 1, Target A	4	3	1
Movie 1, Target C	4	4	0 (swap)
Movie 1, Target T	5	4	1
		$\overline{x}_\delta$	0.33

mean difference. Many such trials need to be performed to estimate the distribution of random trials and test the null hypothesis.

For my small example, the distribution can be determined exactly by exhaustively trying every one of the 2^6, or 64 possible combination. Of the 64 possible combinations only 4 produce a mean difference greater than 0.33. Therefore, the null hypothesis can only be rejected with confidence 0.9375. In other words, 6.25% of the randomly generated trials are greater than the observed mean difference. Intuitively, if a 95% confidence is desired, the CEP values for these two combinations of experts may represent the same distribution.

When the number of test is much larger, such as the 30 or 150 reported in this dissertation, random sampling is used to estimate the distribution of random trials because exhaustively testing all 2^{30} or 2^{150} possible combinations is too computationally demanding. For this dissertation I use 1000 random trials to test the significance of the difference between CEP values measured for various combinations of experts.

C.2 Bonferoni Adjustment

If the same set of results are used for multiple paired sample tests, the simultaneous confidence level will be less than the confidence level for each individual test [Dev87]. For example, assume data set 1 (μ_1) is compared to data sets 2 (μ_2) and data set 3 (μ_3). Also assume the difference between the means, $\mu_1 - \mu_2$ and $\mu_1 - \mu_3$, using the paired sample test, is significant in each case with a 0.950 confidence level. The simultaneous confidence level that μ_1 is statistically different from μ_2 and μ_3 is the product of the confidence levels 0.950^2 or 0.902. This fails to meet the desired 0.950 confidence level.

The Bonferoni adjustment is used to increase the confidence of each individual test so the simultaneous confidence level, or product of the confidence levels, meets the desired 0.950 threshold. Using the previous example, if each individual test confidence level is increased to 0.975, the simultaneous confidence level is 0.975^2 or 0.950.

For many of my tests, I compare the extended condensation filter's combined estimate with two to five contributing experts. In the case where five contributing experts are used, the individual confidence level 0.990 is used so the simultaneous confidence level is 0.990^5 or 0.951. Table C.3 shows the individual thresholds used as a function of the number of contributing experts.

Table C.3: Individual confidence levels required to obtain a simultaneous confidence level of 0.950 using N experts.

Number of Experts (N)	Single Test Threshold
1	0.950
2	0.975
3	0.983
4	0.987
5	0.990

Bibliography

[Bal81] Dana H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[Bar83] S. T. Barnard. Interpreting perspective images. In *Journal of Artificial Intelligence*, volume 48, pages 57–86, 1983.

[Bes89] Paul J. Besl. Active optical range imaging sensors. In J. Sanz, editor, *Advances in Machine Vision*, pages 1–63. Springer-Verlag, 1989.

[Bev93] Ross J. Beveridge. *Local Search Algorithms for Geometric Object Recognition, Optimal Correspondence, and Pose.* PhD thesis, University of Massachusetts at Amherst, 1993.

[BHS91] T. Back, F. Hoffmeister, and H. P. Schwefel. A survey of evolution strategies. In *Proceedings, 4th International Conference on Genetic Algorithms.* Morgan-Kaufmann, 1991.

[Bil97] Jeff A. Bilmes. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-021, U. C. Berkley, 1997.

[Bla92] Andrew Blake. *Active Vision.* The MIT Press, Cambridge, MA, 1992.

[Bor99] Herman Borotschnig. *Uncertain Information Fusion in Active Object Recognition.* PhD thesis, Technischen Universitat Graz, 1999.

[Bre94] Leo Breiman. Bagging predictors. Technical Report TR 421, U. C. Berkley, 1994.

[CA99] Carlo Colombo and Benedetto Allotta. Image-based robot task planning and control using a compact visual representation. *IEEE Transactions on Systems, Man, and Cybernetics*, January 1999.

[Can86] John Canny. A computational approach to edge detection. *Transactions on Pattern Analysis and Machine Intelligence*, pages 679–698, 1986.

[Chu48] Winston Churchill. *The Gathering Storm*, volume 1 of *The Second World War*. London: Cassell and Co., Ltd., 1948.

[Coh95] Paul R. Cohen. *Emperical Methods for Artificial Intelligence.* MIT Press, Cambridge, MA, 1995.

[D$^+$99] Frank Dellaert et al. Using the condensation algorithm for robust, vision-based mobile robot localization. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, June 1999.

[Dev87] Jay L. Devore. *Probability and Statistics for Engineering and the Sciences.* Brooks/Cole Publishing Co., Monterey, CA, 1987.

[DH73] Richard Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[DL00] P. DelMoral and M. Ledoux. On the convergence and the applications of empirical processes for interacting particle systems and nonlinear filtering. *Journal of Theoretical Probability*, 13(1):225–257, 2000.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. R. Statistics Soc. B*, 39:185–197, 1977.

[Ega75] James P. Egan. *Signal Detection Theory and ROC Analysis.* Academic Press, New York, NY, 1975.

[FB80] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In *Proceedings of the Image Understanding Workshop*, pages 71–88, 1980.

[Fog00] David B. Fogel. *Evolutionary Computation.* IEEE Press, New York, NY, 2000.

[FS99] Yoav Freund and Robert E. Schapire. A short introduction to boosting. *Journal of Japanese Society for AI*, pages 771–780, September 1999.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning.* Addison-Wesley Publishing Company Inc., Reading, MA, 1989.

[Gor97] Neil D. Gordon. A hybrid bootstrap filter for target tracking in clutter. *IEEE Transactions on Aerospace and Electronic Systems*, January 1997.

[GSS93] Neil D. Gordon, David J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEEE Proceedings-F*, volume 140, pages 107–113, 1993.

[GZ86] C. Genest and J. V. Zidek. Combining probability distributions: A critique and annotated bibliography. *Statistical Science*, pages 114–148, 1986.

[Hei96] Gary W. Heiman. *Basic Statistics for the Behavioral Sciences.* Houghton Mifflin Co., Boston, MA, 2 edition, 1996.

[Hin99] Geoffrey Hinton. Product of experts. In *Proceedings of the Ninth International Conference on Artificial Neural Networks*, volume 1, pages 1–6, 1999.

[HK92] S. A. Hutchinson and A. C. Kak. Multi-sensing using dempster/shafer theory. In Mongi A. Abidi and Rafael C. Gonzalez, editors, *Data Fusion in Robotics and Machine Intelligence*, chapter 4. Academic Press, 1992.

[HM69] J. Handschin and D. Mayne. Monte carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International Journal of Control*, 9(5):547–559, 1969.

[HR92] Daniel P. Huttenlocher and William J. Rucklidge. A multi-resolution technique for comparing images using the hausdorff distance. Technical Report TR 92-1321, Cornell University, Department of Computer Science, 1992.

[HR97] Timothy Huang and Stuart Russell. Object identification in a bayesian context. In *Proceedings of the 15th International Conference on Artificial Intelligence*, 1997.

[HS96] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision.* Addison-Wesley, Cambridge, MA, 1996.

[IB96] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. *European Conference on Computer Vision (ECCV)*, pages 343–356, 1996.

[IB98] Michael Isard and Andrew Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29, 1998.

[JDM00] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *Transactions on Pattern Analysis and Machine Intelligence*, 22(1), January 2000.

[Kal60] Rudolph E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the American Society of Mechanical Engineers – Journal of Basic Engineering*, pages 35 – 45, March 1960.

[Kan81] Takeo Kanade. Recovery of 3d shape of an object from a single view. *Journal of Artificial Intelligence*, 17, 1981.

[KHDM98] Josef Kittler, Mohamad Hatef, Robert P. W. Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Aerospace and Electronic Systems*, 20(3), March 1998.

[M$^+$99] Bruce Maxwell et al. Using vision to guide an hors d'oeuvres serving robot. In *Proceedings of the Second Workshop on Perception for Mobile Agents.* IEEE, June 1999.

[Mac00] John MacCormick. *Probabilisitc modelling and stochastic algorithms for visual localisation and tracking.* PhD thesis, University of Oxford, 2000.

[Mar82] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* W.H. Freeman and Company, New York, NY, 1982.

[May79] Peter S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, 1979.

[McC81] K. J. McConway. Marginalization and linear optinion pools. *Journal of the American Statistical Association*, 76(374), June 1981.

[Mit97] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[Mur85] Fionn Murtagh. *Multidimensional Clustering Algorithms*, volume 4 of *Lectures in Computational Statistics.* Physica-Verlag, 1985.

[MW95] William Macready and David Wolpert. What makes an optimization problem hard? In *Proceedings, 1995 International Conference on Genetic Algorithms (ICGA-95)*, July 1995.

[Pao94] Lucy J. Pao. Multisensor multitarget mixture reduction algorithms for tracking. *Journal of Guidance, Control, and Dynamics*, November 1994.

[Pea88] Judea Pearl. *Probabilistic reasoning in intelligent systems.* Morgan Kauffman, 1988.

[Pea00] Judea Pearl. *Causality : Models, Reasoning, and Inference.* Cambridge University Press, March 2000.

[RB95] Rajesh P. N. Rao and Dana H. Ballard. An active vision architecture based on iconic representations. *Artificial Intelligence Journal*, 78:461–505, 1995.

[RKMI95] Paolo Remagnino, Josef Kittler, Jiri Matas, and John Illingworth. Camera control for establishing the current and next-look direction in an active vision object recognition system. In J. L. Crowley and H. I. Christensen, editors, *Vision as Process*, chapter 4, pages 403–417. Springer-Verlag, 1995.

[Rub87] Donald B. Rubin. Using the sir algorithm to simulate posterior distributions. In D. Lindley J. Bernardo, M. DeGroot and A. Smith, editors, *Bayesian Statistics 3: Proceedings of the third Valencia international meeting*, pages 395–402. Oxford University Press, June 1987.

[SB91] Michael J. Swain and Dana H. Ballard. Colour indexing. *International Journal of Computer Vision*, 1991.

[SB98] Richard S. Sutton and Andrew G. Bartow, editors. *Reinforcement Learning.* MIT Press, Cambridge, MA, 1998.

[SB01] Mark R. Stevens and J. Ross Beveridge. *Integrating Graphics and Vision for Object Recognition.* Kluwer Academic Publishers, Norwell, MA, 2001.

[Sor70] H. W. Sorenson. Least-squares estimation: from gauss to kalman. *IEEE Spectrum*, 7:63–68, July 1970.

[Swe64] John A. Swets, editor. *Signal Detection and Recognition by Human Observers.* John Wiley and Sons, New York, NY, 1964.

[TFB98] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998.

[TV98] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice Hall, Upper Saddle River, NJ, 1998.

[WG73] William S. Widnall and Peter A. Grundy. Inertial navigation system error models. Technical Report TR-03-73, Guidance Test Division, 6585th Test Group, Air Force Special WEapons Center, May 1973.

[Whi94] Darrel Whitley. A genetic algorithm tutorial. *Statistics and Computing*, 4:65–85, 1994.

[Won98] Chen Pang Wong. An evolutionary optimiser pattern language. In *Proceedings, Midlands UK Patterns Group Meeting*, December 1998.